

GUIDANCE AND CONTROL OF TACTICAL MISSILES

Thomas Alan Grote

NAVAL POSTGRADUATE SCHOOL

Monterey, California



THESIS

GUIDANCE AND CONTROL OF TACTICAL MISSILES

by

Thomas Alan Grote

December 1979

Thesis Advisor:

D. J. Collins

Approved for public release; distribution unlimited

T196176

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Guidance and Control of Tactical Missiles		5. TYPE OF REPORT & PERIOD COVERED Master's Thesis; (December 1979)
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Thomas Alan Grote		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Postgraduate School Monterey, California 93940		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Postgraduate School Monterey, California 93940		12. REPORT DATE December 1979
		13. NUMBER OF PAGES 173
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) Naval Postgraduate School Monterey, California 93940		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
16. DISTRIBUTION STATEMENT (of this Report) Approved for public release; distribution unlimited.		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)		
18. SUPPLEMENTARY NOTES		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) MOD6DF - STM		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) This thesis discusses the conversion of the MOD6DF computer program for use on the IBM-360 computer at the Naval Postgraduate School. The functioning program was modified to investigate the impact miss distance for the Supersonic Tactical Missile. When the initial y-displacement error exceeded 1800 feet, the missile did not acquire the target. All errors smaller than this resulted in miss distances within 0.5 feet of the target. The midcourse guidance reference altitude was changed to reflect a sea-skimming missile. This		

simulation ran and impact was recorded. An attempt at adding random noise to the homing seeker was tried, but revealed that more information is required on this topic. The MOD6DF computer program was successfully converted and altered to run using the simplified ramjet model.

Approved for public release; distribution unlimited

Guidance and Control of Tactical Missiles

by

Thomas Alan Grote
Lieutenant, United States Navy
B.S.A.E., United States Naval Academy, 1974

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN ENGINEERING SCIENCE

from

NAVAL POSTGRADUATE SCHOOL
December 1979

Thesis

G 46075

c.1

ABSTRACT

This thesis discusses the conversion of the MOD6DF computer program for use on the IEM-360 computer at the Naval Postgraduate School. The functioning program was modified to investigate the impact miss distance for the Supersonic Tactical Missile. When the initial y-displacement error exceeded 1800 feet, the missile did not acquire the target. All errors smaller than this resulted in miss distances within 0.5 feet of the target. The midcourse guidance reference altitude was changed to reflect a sea-skimming missile. This simulation ran and impact was recorded. An attempt at adding random noise to the homing seeker was tried, but revealed that more information is required on this topic. The MOD6DF computer program was successfully converted and altered to run using the simplified ramjet model.

TABLE OF CONTENTS

LIST OF TABLES -----	6
LIST OF FIGURES -----	7
TABLE OF SYMBOLS AND ABBREVIATIONS -----	8
ACKNOWLEDGEMENT -----	12
I. INTRODUCTION -----	13
II. MISSION REQUIREMENTS -----	14
A. SEPARATION -----	15
B. MIDCOURSE GUIDANCE -----	18
C. TERMINAL DIVE -----	18
D. TERMINAL GUIDANCE -----	19
III. COMPUTER SIMULATIONS -----	20
A. EXECUTIVE PROGRAMS -----	21
B. OPERATIONAL SUBROUTINES -----	23
C. MODULES -----	24
D. INPUT DATA DECK -----	29
E. AUXILIARY DECK -----	34
F. SEQUENCING -----	36
IV. COMPUTER PROGRAM CHECKOUT -----	40
A. MIDCOURSE GUIDANCE FLIGHT -----	41
B. TERMINAL GUIDANCE FLIGHT -----	47
V. TERMINAL FLIGHT DISTURBANCES -----	56
VI. CONCLUSION -----	64
COMPUTER OUTPUT -----	65
COMPUTER PROGRAM -----	68
LIST OF REFERENCES -----	171
INITIAL DISTRIBUTION LIST -----	173

LIST OF TABLES

I.	TXBA(073) Comparison (MIDCOURSE)	-----	48
II.	VXTP(286) Comparison (MIDCOURSE)	-----	48
III.	YTP(298) Comparison (MIDCOURSE)	-----	49
IV.	TXBA(073) Comparison (TERMINAL)	-----	49
V.	VXTP(286) Comparison (TERMINAL)	-----	50
VI.	YTP(298) Comparison (TERMINAL)	-----	50
VII.	Initial Conditions	-----	57
VIII.	Impact Results	-----	58

LIST OF FIGURES

1. Control Phases/Mission Phases Relationship -----	16
2. Flight Mode Sequencing -----	17
3. Typical Type 2 Card -----	32
4. Typical Type 3 Card -----	32
5. Module Diagram -----	38
6. Processing Sequence -----	39
7. Altitude vs. Time (MIDCOURSE) -----	42
8. Thrust vs. Time (MIDCOURSE) -----	44
9. VXTP vs. Time (MIDCOURSE) -----	45
10. YTP vs. Time (MIDCOURSE) -----	46
11. Altitude vs. Time (TERMINAL) -----	51
12. Thrust vs. Time (TERMINAL) -----	53
13. VXTP vs. Time (TERMINAL) -----	54
14. YTP vs. Time (TERMINAL) -----	55
15. Altitude vs. Time (TERMINAL-ZTP) -----	60
16. Impact Results -----	61
17. Altitude vs. Time (SEA-SKIMMER) -----	62

TABLE OF SYMBOLS AND ABEREVIATIONS

The following is a list of the abbreviations and some of the more common fortran symbols used in the MOD6DF computer program. Each symbol is defined in two ways. The primary source of identification is by its COMMON(3415) location and the second is by its fortran symbol. The fortran symbol is not always a good identifier since it will change from subroutine to subroutine (i.e. W and WE both are used for missile weight).

A. ABEREVIATIONS

<u>Symbol</u>	<u>Definition</u>
TP	tangent plane axes
BA	body axes
SA	stability axes

B. FORTRAN SYMBOLS

<u>Symbol</u>	<u>Definition</u>
TXEA(073)	missile thrust in the x-direction in BA (lb)
TYEA(074)	missile thrust in the y-direction in BA (lb)
TZEA(075)	missile thrust in the z-direction in BA (lb)
W(086)	missile weight (lb)
S(110)	missile reference area (ft ²)
CA(111)	drag coefficient
CY(113)	side-force coefficient
CZ(115)	normal force coefficient
CBAR(116)	mean aerodynamic chord (ft)
CMQ(118)	damping in pitch coefficient
CNR(119)	damping in yaw coefficient

<u>Symbol</u>	<u>Definition</u>
CLP(120)	damping in roll coefficient
CM(121)	pitching moment coefficient
CN(122)	yawing moment coefficient
CL(123)	rolling moment coefficient
CGI(136)	center of gravity (ft)
A(201)	moment-of-inertia about missile roll axis (x-body axis) (slug-ft ²)
B(202)	moment-of-inertia about missile pitch axis (y-body axis) (slug-ft ²)
CC(203)	moment-of-inertia about missile yaw axis (z-body axis) (slug-ft ²)
TSA(208)	angle between SA and EA (rad)
P(212)	missile angular velocity about x-BA (rad/s)
Q(216)	missile angular velocity about y-BA (rad/s)
R(220)	missile angular velocity about z-BA (rad/s)
AG(282)	unit conversion lb-slug
VXTP(286)	missile velocity in x-TP (ft/s)
XTP(290)	missile displacement in x-TP (ft)
VYTP(294)	missile velocity in y-TP (ft/s)
YTP(298)	missile displacement in y-TP (ft)
VZTP(302)	missile velocity in z-TP (ft/s)
ZTP(306)	missile displacement in z-TP (ft)
GZRO(404)	constant set to zero for flat earth gravitational field and set to one for a spherical gravitational field
ER(405)	angular velocity of earth (rad/s)
ALPO(406)	angle between north and x-TP (rad)

<u>Symbol</u>	<u>Definition</u>
OLAMO(407)	latitude origin of tangent plane
HO(414)	distance tangent plane is from earth (ft)
GO(415)	gravitational acceleration (ft/s ²)
HREF(501)	reference altitude for midcourse guidance (ft)
RE(503)	earth's radius (ft)
H(507)	altitude normal to earth (ft)
AMACH(520)	missile Mach number
THET(521)	missile pitch angle, TP (rad)
PSI(522)	missile yaw angle, TP (rad)
PHI(523)	missile roll angle, TP (rad)
GAMMAV(527)	vertical flight path angle, TP (rad)
VEL(528)	magnitude of missile velocity (ft/s)
VAT(529)	missile velocity (ft/s)
TF(550)	program termination time (s)
VAH(561)	computed speed of sound (ft/s)
ALAT(576)	latitude of target position (deg)
AZ(578)	azimuth of target position (deg)
DYNP(581)	dynamic pressure (psi)
FLGRJ(606)	constant set to zero indicates run will use ENGINE subroutine and when set to one run will use RAMJET subroutine (simplified ramjet model)
T(932)	actual time (s)
T1(933)	boost engine ignition (s)
T2(934)	commence acceleration command mode (s)
T3(935)	boost engine burn-out/port cover blow-in (s)
T4(936)	start ramjet engine (cruise) (s)
T5(937)	commence heading and altitude guidance control (s)

<u>Symbol</u>	<u>Definition</u>
T6(938)	commence terminal dive (s)
T7(939)	commence terminal guidance - search (s)
T8(940)	commence terminal guidance - track (s)
DMAX(1740)	maximum fin deflection (rad)
CPP(2669)	time between printouts (s)
ROLLO(2901)	initial roll angle (rad)
PITCHO(2902)	initial pitch angle (rad)
YAWO(2903)	initial yaw angle (rad)
STEP(2905)	determines executive program flow after staging
DOC(2909)	defines number of times COMMON will be printed
HMIN(2911)	minimum integration step size
HMAX(2912)	maximum integration step size
DER(1)(2913)	integration step size (s)

ACKNOWLEDGEMENT

This author is grateful to Professor D. L. Collins, NAVAL POSTGRADUATE SCHOOL, for his helpful discussions and suggestions. I also acknowledge NWC CHINA LAKE for their cooperation in obtaining the MOD6DF computer program.

I. INTRODUCTION

Research was undertaken to convert the MOD6DF computer program received from NWC China Lake for use on the IBM-360 computer at the Naval Postgraduate School. Once a functioning program was obtained, initial displacement error versus impact miss distance was investigated for the supersonic tactical missile. Additionally, the midcourse guidance reference altitude was changed to reflect a sea-skimming scenario and this was examined for its effect on the terminal guidance problem. This report not only discusses the aforementioned topics, but also describes the missile mission requirements and the MOD6DF computer program.

II. MISSION REQUIREMENTS

The Supersonic Tactical Missile (STM) mission is divided into six phases (Ref. 1).

- * initial conditions
- * separation
- * boost
- * transition
- * cruise
- * terminal

These divisions are based on the missile aerodynamics. The initial condition phase establishes the starting conditions for each launch. This is done while the missile is still attached to the launch platform. The separation phase starts when the missile is launched. The missile falls for approximately five seconds until the boost engine ignites. This initiates the boost phase which continues until the missile achieves Mach two. As the missile passes through Mach one, plume effects are encountered which the aerodynamics account for. At the end of the boost phase, the port covers blow in. This initiates the transition phase. This phase is very short and allows the debris to be ejected from inlet ports. The cruise phase commences when the ramjet engine ignites. This engine propels the missile until target impact. The terminal phase of the flight begins when the missile is commanded to dive from the cruise altitude. This phase concludes when the flight is terminated at target impact.

The six STM mission phases require four phases of control. These control phases are:

- * separation
- * midcourse guidance
- * terminal dive
- * terminal guidance

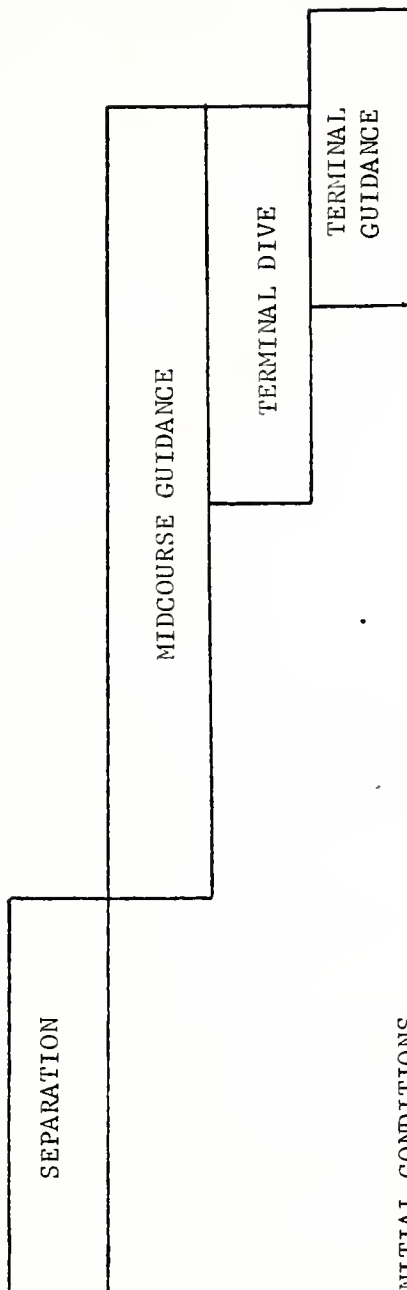
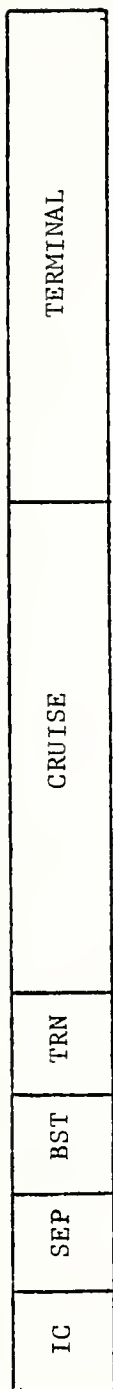
The four Guidance, Navigation, and Control (GN&C) system control phases are directly related to the mission phases. Figure 1 displays this relationship (Ref. 2). To obtain a successful flight, various types of control and guidance functions are required. Figure 2 (Ref. 3) illustrates the guidance control mode sequencing in relation to the mission phases and also indicates the critical missile switching times. The control phases are discussed below, along with the appropriate guidance modes.

A. SEPARATION

The separation phase commences upon launch. The missile is ejected downward from the launch platform. Additionally, the pitch attitude of the missile is commanded down. Since this portion of the flight is unguided, the ejection force and gravity are the only forces acting on the missile. At launch, the missile is required to be in the attitude command mode.

Five seconds into the flight, the missile pitch attitude is commanded up and the boost engine is ignited. The booster continues until the missile attains Mach two, at which time control is shifted from attitude to acceleration control mode. The acceleration control then requires the missile to maintain the normal and lateral accelerations at zero.

The last event to occur in the separation phase is inlet port cover blow-in. The time delay that allows the port covers to clear is a function of altitude.

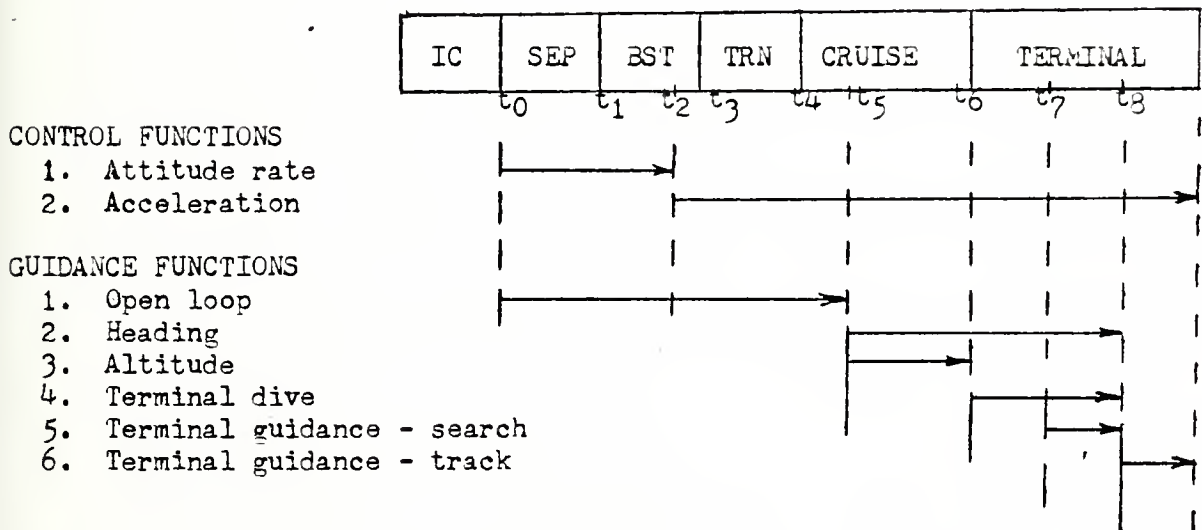


IC - INITIAL CONDITIONS
 SEP - SEPARATION
 BST - BOOST
 TRN - TRANSITION

CONTROL PHASES/MISSION PHASES
RELATIONSHIP

FIGURE 1

- t₀ - LAUNCH: START SIMULATION
- t₁ - BOOSTER IGNITION
- t₂ - CONTROL SYSTEM MODE CHANGE FROM ATTITUDE TO ACCELERATION CONDITION AT M=2.0
- t₃ - PORT COVER BLOW-IN, BOOSTER BURNOUT
- t₄ - RAMJET IGNITION, t₃ + .3 sec
- t₅ - ENGAGE GUIDANCE MODES
- t₆ - DIVE COMMAND
- t₇ - SEEKER SEARCH MODE IS ACTIVATED
- t₈ - TERMINAL TRACKING



FLIGHT MODE SEQUENCING

FIGURE 2

B. MIDCOURSE GUIDANCE

Midcourse guidance begins upon completion of the separation phase. The moment the port covers are clear, the phases shift. During this phase two things are required. The GN&C must guide the missile to the established cruise altitude and then guide the missile (in the horizontal plane) to the predetermined target location.

To accomplish these requirements the GN&C system employs both altitude and heading control. The altitude control acts upon the pitch axis to drive the missile to the cruise altitude and then to maintain that altitude until the terminal dive phase commences.

The actual guidance work is performed by the yaw axis. The GN&C system uses the heading control to steer the missile to the target position. To generate the steering commands a guidance law is necessary. The guidance law should minimize the cross-track error and the final lateral acceleration. A minimum cross-track error allows for minimum flight time and minimum displacement error when target search is initiated. Minimizing the final lateral acceleration ensures that the seeker will continuously be pointing toward the target area. To insure accuracy, the guidance law must be able to perform these functions when subjected to disturbances such as wind and thrust misalignment.

C. TERMINAL DIVE

The terminal dive phase commences when the missile is commanded to dive. The actual time that this command is given is a function of the predetermined target coordinates. Upon diving, the missile must accelerate downward along a 60-degree (from horizontal) dive angle to the target position. Once the missile has steadied up in the proper dive aspect, terminal guidance commences.

D. TERMINAL GUIDANCE

During the terminal guidance phase the seeker is required to search, acquire, and track the target. The GN&C system then uses these tracking signals to direct the missile to the target. The seeker uses a preprogrammed search pattern to locate the target. When the target has been acquired, the seeker then commences tracking it and also notifies the GN&C system that the target is being tracked.

Target acquisition takes place when the target falls within the instantaneous four-degree beamwidth pattern as it traverses the scan pattern on the earth's surface. The computer program uses this assumption since the exact missile target acquisition mechanism has not yet been defined.

Upon acquisition, the GN&C system closes the seeker tracking loop. The radiometer error signal is defined as the difference between the look vector and the line-of-sight. This signal is used to reposition the antenna gimbals so the look vector and the line-of-sight coincide. Under the closed loop operation the radiometer output is proportional to the line-of-sight rate and this is the signal used to guide the missile to target impact.

III. COMPUTER SIMULATION

The modularized six-degree-of-freedom (MOD6DF) computer program was developed by the Litton Systems, Inc., Guidance and Control Systems Division (Ref. 4) to be used in analyzing missile guidance and control. The program uses a building block approach, where each module corresponds either to a missile subsystem or an environmental system. In its original form, the program is used primarily for terminal guidance of air-to-surface missiles. The Naval Weapons Center, China Lake modified the original program to specifically apply to the Supersonic Tactical Missile (STM). The modification also allows the user the capability of simulating any portion of the missile flight trajectory.

The MOD6DF computer program consists of four main decks and one auxiliary deck. The main decks include the executive programs, operational subroutines, modules, and input data deck. All the subprograms utilized in the integration and the sequencing of the modules are contained in the executive programs. The operational subroutines are used by the user to control the program while it is running. The psysical system and the environmental subroutines are included in the modules. The input data deck contains all the information the executive programs need to execute the desired run. Lastly, the auxiliary deck consists of all those subprograms (subroutines and functions) that are required by the modules.

Once the user starts making simulations, he must concern himself with program sequencing. Proper sequencing is required to ensure a valid run is achieved. Since it is of such importance, sequencing will be discussed as the final part of this section.

A. EXECUTIVE PROGRAMS

The executive programs are the main core of the MOD6DF program. These subroutines have various functions in setting up, sequencing, integrating, and resetting the program. Since these functions are required by any type of analysis done, the user should never have to change any of these subroutines. Reference 5 should be consulted if the user desires more information about the content of any of the subroutines described herein.

1. Zero

Subroutine ZERO is used to set all the COMMON(3415) locations to the value zero. This is done to ensure that no erroneous information is used in the simulation.

2. Oinpt1

Subroutine OINPT1 is the basic input routine for the MOD6DF program. The normal input is from punched cards. However, inputs may also be read from tapes. The basic input cards will be discussed in the section covering the input data deck.

3. Auxi

Subroutine AUXI is used to call the initialization modules for each run. Additionally, it sets up the list of state variables which are used in AMRK.

4. Auxsub

Subroutine AUXSUB is used to call the dynamic modules. In calling the dynamic modules, AUXSUB sets and resets the lists needed for AMRK and the COMMON(3415) storage cells with the most recent values of the state variables and their derivatives.

5. Amrk

Subroutine AMRK is the integration subroutine. It uses a point-wise first order Runge-Kutta method. All the state variables to be integrated must be listed in COMMON(3415) and also must appear in a processing list.

6. Reset

Subroutine RESET is used to reinitialize up to fifty input parameters. This is done prior to the start of any repeated runs. To indicate which parameters are to be reset, the number one is punched in columns 46-60 on the respective type three card.

7. Return Group

The return group is a collection of all the unused modules. Since all the modules and their initialization modules are called by AUXI and AUXSUB, the unused ones must still remain in the deck. These are required to ensure proper linking when the computer attempts to link all the subroutines. All the subroutines in this grouping contain three cards. The three cards are the subroutine title, and a return and an end card.

8. Subl1, Subl2, Subl3

These subroutines are used to call the operational subroutines that are required. They call the routines in the order prescribed by the input data deck. The number at the end of each subroutine title indicates which operational subroutines it can call. For example, SUB11 can call STGE1.

B. OPERATIONAL SUBROUTINES

The operational subroutines provide the user with control of the program while it is running. The order in which the operational subroutines are called is specified by the input data deck. Since these routines assist the user in controlling the simulation, they can be reprogrammed. However, it is advised that they not be changed until the user has become quite familiar with the overall operation and sequencing of the MOD6DF program. For more in depth knowledge of the operational subroutines the user should consult Ref. 6, and the computer listing, which is contained at the end of this report.

1. Inpt1, Inpt2, Inpt3

These subroutines are available for new inputs during the simulated flight. The only one presently used in the program is INPT1. It is utilized to input a namelist file which is used by ENGINE and is described there.

2. Oupt1, Oupt2, Oupt3

These subroutines allow for the print-out of up to fifty different variables during the flight simulation. OUP1 is not utilized in the deck. OUP2 is used when the desired output is to be put on tape. OUP3 is the basic output routine. It prints the desired output on regular computer paper.

3. Stge1, Stge2, Stge3

These subroutines allow for proper staging, run termination, etc.. Presently STGE1 is not being used. STGE2 is being used as the staging initialization subroutine. STGE3 is the primary subroutine of this group. It stages when impact with the earth is made, when the final time, TF(550), is reached, or when LCONV(2672) is set equal to two. All the tolerances for staging are listed in STGE3.

4. Cntr1, Cntr2, Cntr3

These subroutines allow the external dynamic control inputs to the modules. In the MOD6DF program these are not used.

5. Rndm1, Rndm2, Rndm3

These subroutines allow random noise to be added to the state variables generated in the modules. RNDM1 is not used in the program. RNDM2 is used as the initialization subroutine, while RNDM3 provides continuous noise values. These subroutines produce correlated noise values for as many modules as required. The noise values remain fixed during each individual integration cycle.

6. Auxa1, Auxa2, Auxa3 Auxb1, Auxb2, Auxb3 Auxc1, Auxc2, Auxc3

These subroutines are auxiliary routines that allow for external input, output, control, etc. of the modules. At the present time, none are utilized in the MOD6DF program.

C. MODULES

The modules are of prime importance to the user since they represent the 'model' of the dynamic system. In general, the model is described by ordinary non-linear time-varying differential equations with both random and deterministic forcing functions. The user must first reduce these equations to an equivalent system of first order equations, which can then be described by each module. Generally speaking, the physical system is so complex that this would be impossible to do. However, due to the modularity of the MOD6DF program, the user can think of each module as a completely independent system described by the equations within that module.

There are thirty-six possible modules divided into five functional

categories. Each group is identified by a letter which pertains to that group's function; A (airframe), C (computers), D (dynamics), G (geophysical), S (sensors). A complete printing of each module is contained in the computer program listing.

1. Airframe

a. Subroutine A1

This is the aerodynamic forces and moments module. It calculates all the necessary forces and moments in body axes. These values are then used in the computation of the dynamics.

b. Subroutine A2

This is the missile aerodynamic coefficient module. It calculates the required coefficients using the information stored in the BLOCK DATA. Using the timing inputs, this routine computes the coefficients for the different effects. Some of the effects accounted for are; plume, separation, and control surfaces effectiveness. With this done the total coefficients are determined.

c. Subroutine A3

This is the missile propulsion module. The timing inputs are used to determine whether the missile is in free fall, boost, transition, or cruise phase. With this determined the correct engine subroutine (BOOST, RAMJET, ENGINE) can be called. Three variables are calculated using the body axes as the frame of reference. They are the missile thrust in all coordinate directions, the principal moments of inertia, and the missile weight.

d. Subroutine A4

This is the fin actuator module. The four control surfaces commands are calculated as either ideal actuators or as second-order

ones. In addition to control surfaces commands, the rate of change of yaw, and roll are computed.

e. Subroutine A5

This module is part of the return group.

2. Computers

a. Subroutine C1

This is the autopilot module for the STV-G. It uses the cruise engine ignition time to divide the routine into boost/separation and cruise phases. These two phases use different algorithms to calculate the turning moments in pitch/yaw and roll.

b. Subroutine C2

This is the guidance command module. Using the timing inputs, this routine is divided into separation/boost, dive/climb, cruise, terminal dive, and terminal homing sections. Each section uses slightly different algorithms to calculate the guidance commands to maintain the proper flight profile.

c. Subroutines C3 - C10

These modules are part of the return group.

3. Dynamics

a. Subroutine D1

This is the translational dynamics module. It computes the total acceleration in body axes and then converts them to the tangent plane reference. Then, accounting for aerodynamics, thrust, gravity, and coriolis, the velocity and acceleration are calculated.

b. Subroutine D2

This is the rotational dynamics module. With the principal axes as a reference, this subroutine computes the body angular rates and the attitude direction cosines.

c. Subroutines D3 - D5

These modules are part of the return group.

4. Geophysical

a. Subroutine G1

This is the gravitational and coriolis acceleration module.

It calculates the gravitational acceleration using one of two fields.

The user specifies the field to be used by an input card. To use a flat-earth gravitational field, GZRO(404) must equal 0.0, and to use a spherical gravitational field, GZRO(404) must equal 1.0.

b. Subroutine G2

This module is part of the return group.

c. Subroutine G3

This is the air data module. It computes the velocity, in all three coordinate directions, with respect to the air mass. These values are then resolved into body and stability axes. This module also computes all the properties of air by calling subroutine AIR. These values are stored in their COMMON(3415) locations for use in the other modules.

d. Subroutine G4

This module is part of the return group.

e. Subroutine G5

This is the coordinate conversion module. It takes the missile position, does a coordinate conversion and then it determines the position, velocity, and acceleration in the ECI system.

f. Subroutine G6

This module is part of the return group.

5. Sensors

a. Subroutine S1

This is the homing seeker module. It simulates the missile seeker and computes the seeker dynamics and Euler angle rates. Several flags are used to control the seeker sequencing:

- (1) FLAGS(335). FLAGS signals the start of the seeker search.
- (2) FLGT(317). FLGT signals when the seeker is locked-on.
- (3) FLGTS(347). FLGTS signals the end of search.
- (4) FLGD(336). FLGD signals when the seeker has detected the target.
- (5) FLAGLT. FLAGLT signals when the target is outside the seeker field of vision.

b. Subroutine S2

This is the radiometer module. It takes the target position and the ATIGS target position and converts them from body axes to the seeker axes. Using target position, the module then calculates the azimuth and elevation error signals.

c. Subroutines S3, S4

These modules are part of the return group.

d. Subroutine S5

This is the accelerometers and gyros modules. The user has the option of using ideal or digital accelerometers. To specify the type of accelerometer, the user must include the appropriate data statement in the subroutine. If ideal accelerometers are desired, FLGA must equal 0.0 and for digital accelerometers, FLGA must equal 1.0.

e. Subroutines S6 - S10

These modules are part of the return group.

D. INPUT DATA DECK

The input data deck provides the user with the means of specifying which operational subroutines and modules are to be utilized for the desired run. It also allows the user to set the starting conditions. In general, only the constants and the state variables must be given initial values. All quantities in COMMON not given initial values will be set to zero by ZERO. In addition to the state variables, the upper and lower error bounds must be initialized. There are seven types of input cards, each indication a certain function.

<u>Type</u>	<u>Function</u>
0	read/write tape
1	operational subroutine to be called
2	module to be called
3	numerical input
4	printed output
5	parameter square and sum
6	termination and random noise generator input

A separate card is required for each subroutine, module, input, and output quantity. A sample computer printout of the input data deck is contained in the computer output section.

1. Type 0 Card

Type 0 cards are used to indicate if the type 3 inputs are to be read from or written onto an auxiliary tape. These procedures can be used rather than reading the inputs from a deck of cards. A typical card is defined by punching a zero in column 2. The field that covers columns 5 - 20 is used by the user for any descriptive statements with which he wishes to identify the input. Column 21 -25 contains the right-

justified integer number of the tape transport to be used. The number of the first record to be read is punched in column 31 - 45. The last field is column 46 - 60 which contains the number of records to be read. The last two fields may use either fixed or floating-point notation.

2. Type 1 Card

Type 1 cards are used to specify which operational subroutines are called during the flight simulation. This type card is identified by the number one in column 2. The second field is column 5 - 20 which contains any identifying information. This information is printed out when the data deck is read and allows the user to read exactly which subroutines were called. Column 21 - 25 contains the right-justified integer number which is the subroutine identifying number. The operational subroutine numbers are;

<u>Subroutine</u>	<u>Subroutine Number</u>
INPT1, INPT2, INPT3	2
OUPT1, OUPT2, OUPT3	3
STGE1, STGE2, STGE3	4
CNTR1, CNTR2, CNTR3	5
RNDM1, RNDM2, RNDM3	6
AUXA1, AUXA2, AUXA3	7
AUXB1, AUXB2, AUXB3	8
AUXC1, AUXC2, AUXC3	9

It should be noted that all or any of the subroutines listed under one number can be called by including only one card. The cards are placed in the data deck in the order in which they will be called.

This order or sequencing will be explained further in section F, Sequencing. A typical type 2 card is identified by the number two in

column 2. In general, column 5 - 20 should contain the module title, but any pertinent information is allowed. The module number is punched in column 21 - 25 and it must be right-justified. A listing of the module numbers follows:

<u>Module</u>	<u>Module Number</u>	<u>Module</u>	<u>Module Number</u>
A1,A1I	2	D4,D4I	20
A2,A2I	3	D5,D5I	21
A3,A3I	4	G1,G1I	22
A4,A4I	5	G2,G2I	23
A5,A5I	6	G3,G3I	24
C1,C1I	7	G4,G4I	25
C2,C2I	8	G5,G5I	26
C3,C3I	9	G6,G6I	27
C4,C4I	10	S1,S1I	28
C5,C5I	11	S2,S2I	29
C6,C6I	12	S3,S3I	30
C7,C7I	13	S4,S4I	31
C8,C8I	14	S5,S5I	32
C9,C9I	15	S6,S6I	33
C10,C10I	16	S7,S7I	34
D1,D1I	17	S8,S8I	35
D2,D2I	18	S9,S9I	36
D3,D3I	19	S10,S10I	37

Note that either the module, the initialization module, or both may be called by including only one card in the deck. A sample of a typical type 2 card is shown in Figure 3 (Ref. 7).

COLUMN

2	5	20	21	25
2	MODULE 55			32

TYPICAL TYPE 2 CARD

FIGURE 3

2	5	20	21	25	31	45	46	60
3	WEIGHT			86	1152.0		1.0	
					1.152	E+03	1.0	E+00

TYPICAL TYPE 3 CARD

FIGURE 4

4. Type 3 Card

Type 3 cards are used to set any COMMON(3415) location to any value other than zero. In general, four items must be initialized. The state variable initial values and any constants used in the flight simulation are the most obvious. Additionally, there are some constants associated with the executive programs and operational subroutines and the state variable upper and lower bounds which must be initialized. As with all cards, column 2 defines the type card and it must contain a three. Column 5 - 20 holds the statement describing the input. The user should be specific here since it will save him having to remember every COMMON(3415) location. The only other means of input identification is by column 21 - 25. These columns contain the right-justified COMMON location of the input. Columns 31 - 45 hold the actual numerical value of the input. The last field, column 46 - 60, contains the reset flag. If the reset flag equals one, the COMMON(3415) location and the numerical value are placed in the reset list. This list may contain up to fifty different variables. This, in the case of multiple runs, allows the variables to be reset to its initial value prior to each run without additional input cards. The sample card in Figure 4 (Ref. 8) shows that either fixed or floating-point notation may be used to input the numerical value and the reset flag. These cards need not be inputted in any specific order, but for ease of checkout, it is advised to place them sequentially by their COMMON location.

5. Type 4 Card

The MOD6DF program can printout a maximum of fifty variables for each simulation. Type 4 cards are used to specify which variables are to be printed. Column 2 must contain the number four to indicate a type

4 card. When the results are printed out headings are included. These headings are designated in column 9 - 20. The exact alphanumeric title punched will be printed at the top of each page, this need not be the fortran symbol used within the program. The COMMON(3415) location of the output variable is contained in column 21 - 25 and must be right-justified.

6. Type 5 Card

Type 5 cards are used to indicate which variables are to be root-mean-squared. These cards are similar to type 1 and type 2 cards. Column 2 must contain the number five. Any pertinent information about the variable to be operated on is punched in column 5 - 20. The COMMON(3415) location of the variable must be right-justified in columns 21 - 25. The last field, column 31 - 45, indicates whether the root-mean-square operation will occur along the trajectory or at the end.

7. Type 6 Card

The type 6 card has two purposes. Its first function is trivial, but required. The number six is typed in column 2 and the rest is blank. This indicates to the computer program that there are no more input cards. Its second function involves random inputs. This card is used to indicate the number of random process (noise) generator cards that are to be read before the input process is terminated. Column 2 has the same information as before. Any pertinent information is contained in column 5 - 20. Columns 21 - 25 must be right-justified and they contain the number of random generator cards to be read.

E. AUXILIARY DECK

The auxiliary deck is a collection of subroutines and functions required by the modules. These routines are general in nature since

they can be called by several modules. They are used to calculate such things as the properties of air, engine performance, various ratios, and to locate values in the many data tables. A brief discussion of the most common subroutines is presented.

1. Boost

This subroutine calculates the thrust coefficient (CT) and the fuel flow rate (FF) during the boost phase. These values are returned to module A3 and used to calculate the missile thrust in the body axes and the missile weight.

2. Ramjet

This subroutine is used during the midcourse, cruise, phase of the missile flight. It uses a simplified ramjet model to calculate the thrust coefficient and fuel flow rate. This routine is not automatically called. The user must designate that he wishes to use it by inputting a type 3 card setting FLGRJ(606) equal to one. This then sets up the proper stepping in module A3.

3. Engine

This is the primary routine during the cruise phase. It is one of many routines within the NWC air-breathing propulsion package. This entire package is utilized to calculate the engine parameters from inlet to exhaust. Again, thrust coefficient and fuel flow rate are eventually computed and returned to module A3. The user does not need to supply any special input cards to use this subroutine. If no initial value is inputted for FLGRJ(606) it is automatically set to zero, which indicates this routine is to be used.

4. Air

All the properties associated with the air are calculated in

this subroutine. This includes the computation of the speed of sound and the dynamic pressure. Since the missile does vary in altitude, the routine takes this into account as well as the latitude. Once these values are calculated, they are returned to module G3.

5. Block Data

This routine contains data tables. These tables cover parameters from aerodynamic coefficients to thermal properties. The total data package covers specific bands within the missile operating envelope. Data is stored in matrices which includes one, two, and three dimensional ones. These data tables are readily available and there are routines designed to retrieve this information quickly.

6. Serch

This subroutine, along with several functions, is used to retrieve information from BLOCK DATA. If the present operating point of the missile is not within one of the bands of information, then the tables are interpolated. The functions THREDL, STDLI, STDLIA, and TAB do the interpolation of the tables. Since the tables are of various lengths, these functions are very general.

F. SEQUENCING

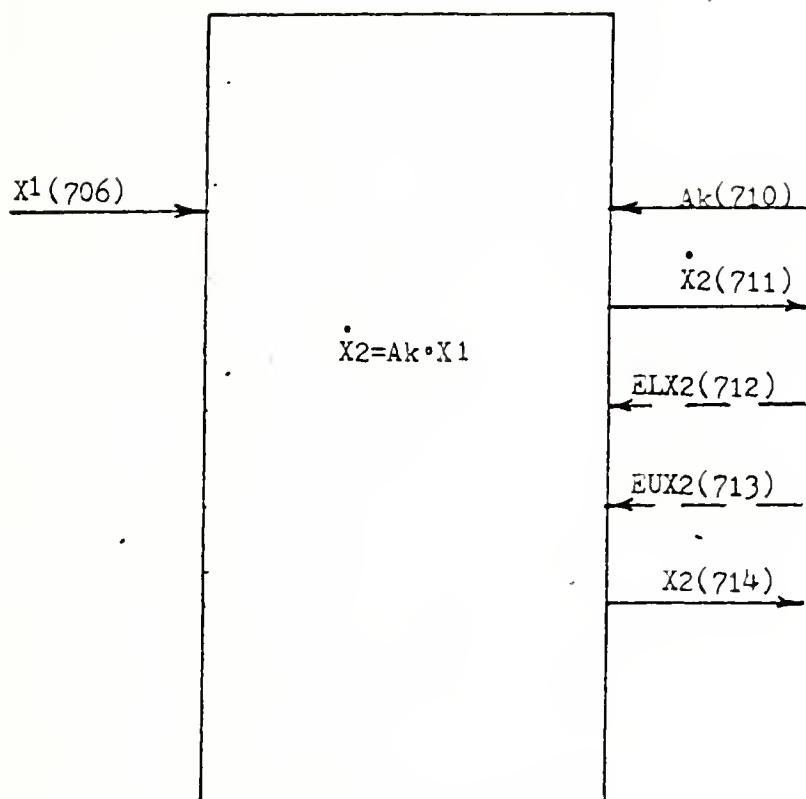
Sequencing is very important in the running of the MOD6DF program. Care must be taken to ensure that the modules are processed in the correct order at each step. This is essential to eliminate the use of obsolete values from the last cycle. An exception to this problem is the state variables. These are updated simultaneously by the integration algorithm. Any module is capable of using the most current value of these, no matter what the order of processing.

To help remedy this problem, module diagrams were devised. Module

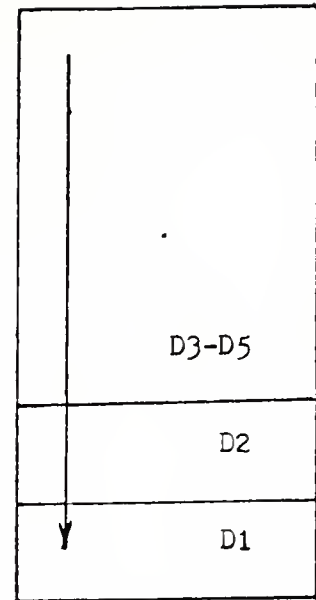
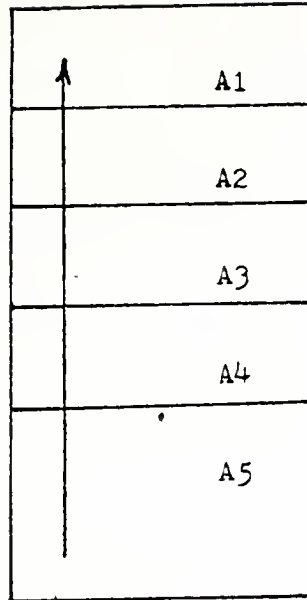
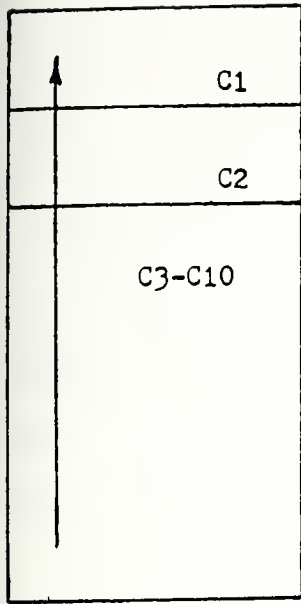
diagrams aid the user in maintaining the proper flow of variables into and out of the modules. To design a module diagram start with a box. This box will contain all the equations for a specific module. The standard procedure for showing inputs and outputs is to use arrows pointing in or out from either the left or the right side. The example in Figure 5 (Ref. 9) shows this technique. The arrows pointing in from the left indicate variable inputs from other modules. Arrows pointing out to the right indicate output going to either other modules or program output. The last set of arrows points in from the right. These show the constants brought in directly or indirectly from the initialization module. Since each arrow represents a variable, they must be defined. The usual means of labeling the arrows is to use the variable fortran symbol and in parenthesis its COMMON(3415) location.

Each variable usually has only one COMMON location associated with it. In the case of 'state' variables this is not true. State variables are defined by four consecutive COMMON locations. The first is for the derivative of the variable. The second and the third locations hold the lower and upper bounds, respectively, of the integration error. The last one contains the variable itself.

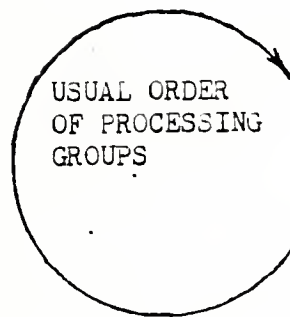
Once all the required module diagrams have been completed, they can be combined to get the overall processing order. The usual processing order, shown in Figure 6 (Ref. 10) is to start with the geophysical group, then proceed to sensors, computers, airframe, and finally dynamics. Within each group is a usual processing order and this too is shown in Figure 6. This process for determining the program sequence will eliminate the use of any obsolete values in the computer run.



MODULE DIAGRAM
FIGURE 5

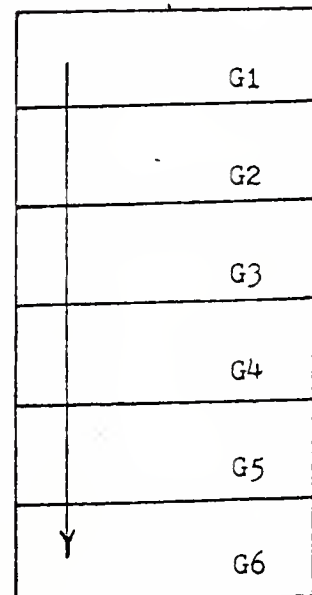
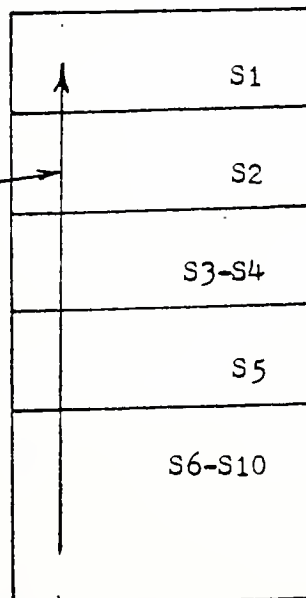


END CYCLE



BEGIN CYCLE

USUAL
ORDER OF
PROCESSING
MODULES



PROCESSING SEQUENCE

FIGURE 6

IV. COMPUTER PROGRAM CHECKOUT

The basis for the research was the MOD6DF computer program from NWC China Lake. The total package received from them consisted of a listing of the program and an uninterpreted deck of cards. The initial step was to input the cards in small groups into the computer and then to examine the source listing. This listing revealed that the original deck was punched in BCD. This fact was easy to determine since several characters were changed (Ref. 11). The library routine NEWDEK was used to translate the cards from BCD to EBCDIC.

Once the translation was completed, an attempt was made to compile the new deck. This produced an output which contained many syntax errors. These errors were divided into two major groups. One effected the use of quotation marks in FORMAT and comment statements. The other one, the more difficult, effected the DATA statements in the BLOCK DATA subroutine.

The problem with the DATA statements was due primarily to the difference in the compilers used. The compiler at NWC was much newer and allowed for the use of more sophisticated inputs. The compiler at NPS only allows a data set to start with the first element. This required the rewriting of many data groups. To complicate these revisions, a limit of nineteen continuation cards is also imposed. These restrictions demanded not only the rewriting of many data sets, but also the formation of two new ones.

With the corrections finally completed, the computer would then compile the program. The next step was to link all the subroutines together. The first attempt was unsuccessful. Inadvertently, the subroutine INTR20 had been omitted from the original deck. Using the program listing, the contents of INTR20 were typed and included in the

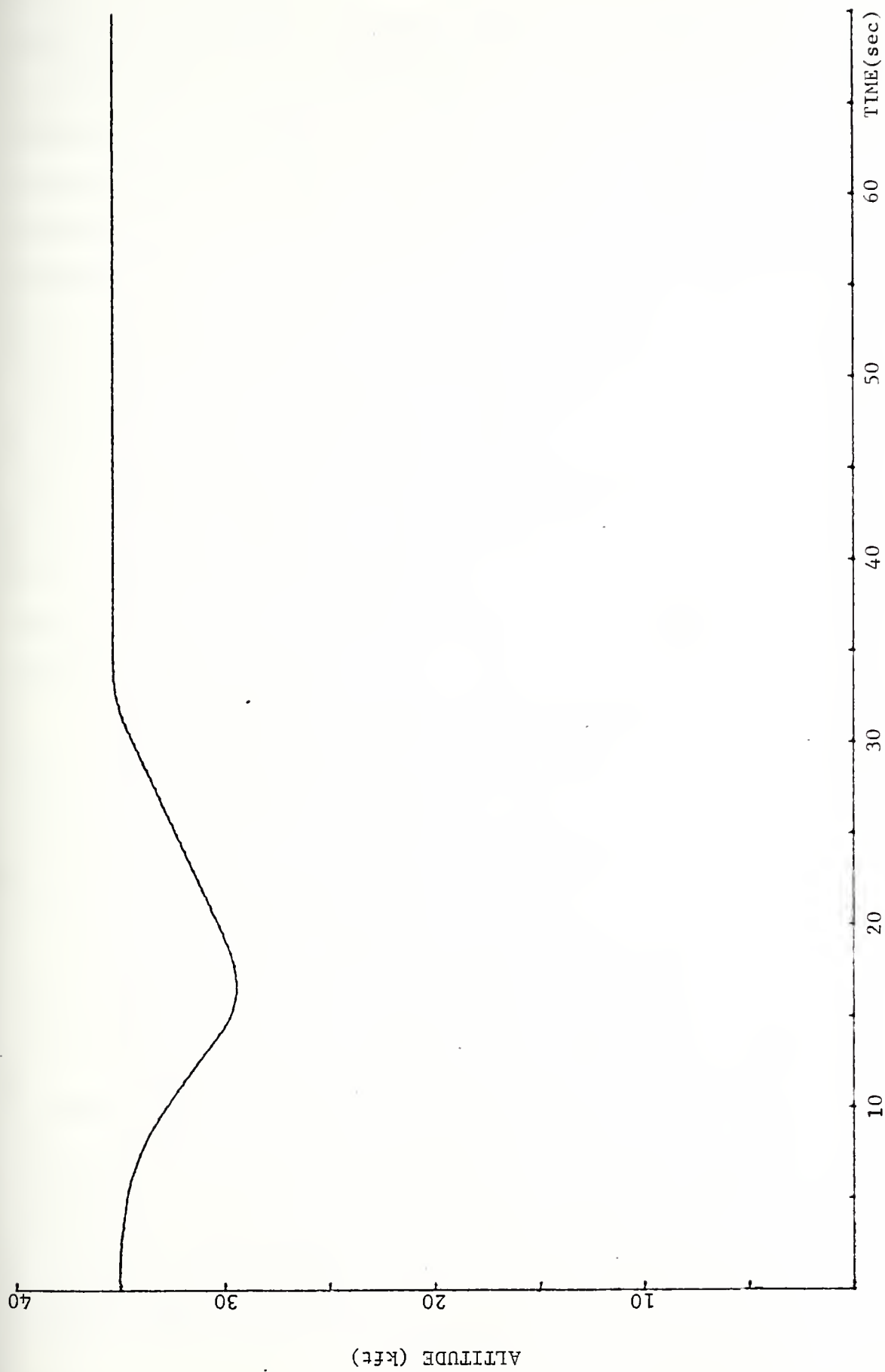
main deck. With this addition the program would now compile and link.

Now that the program would compile and link, attention was turned to getting a good simulation. To facilitate this process, two sample outputs were obtained from NWC. These outputs cover one missile flight which is broken down into a midcourse guidance and a terminal guidance simulation. Hereafter these will be referred to as midcourse baseline and terminal baseline, respectively. Using the initial conditions from the baseline models, it was hoped that the outputs could be duplicated.

A. MIDCOURSE GUIDANCE FLIGHT

The initial run, using the midcourse baseline inputs, revealed an overflow problem with the dynamic pressure (QD(508)). Using the traceback procedures outlined in the Users Manual from the W. R. Church Computer Center, the problem was confined to subroutine AIR. The problem turned out to be a translation error. The symbol $P\emptyset$ (\emptyset - zero) had translated to $P\emptyset$ in one place and P0 in another. This error caused the program to use a value left in that memory location from a previous run to calculate the dynamic pressure. With this problem remedied, the program could progress a little farther. The next stumbling block appeared as a divide check. These errors were resolved by introducing patches that would bypass a statement that tried to divide by zero. Once bypassed, that quantity would be set equal to zero. This was the normal procedure of the computer, but it would stop the run after ten such errors. Having corrected all these errors, output was obtained which covered the desired seventy seconds of flight.

When the output was examined a major switching problem within subroutine A3 (missile propulsion module) was found. The midcourse baseline utilized a simplified ramjet model, but the output was not.



ALTITUDE VS. TIME (MIDCOURSE)

FIGURE 7

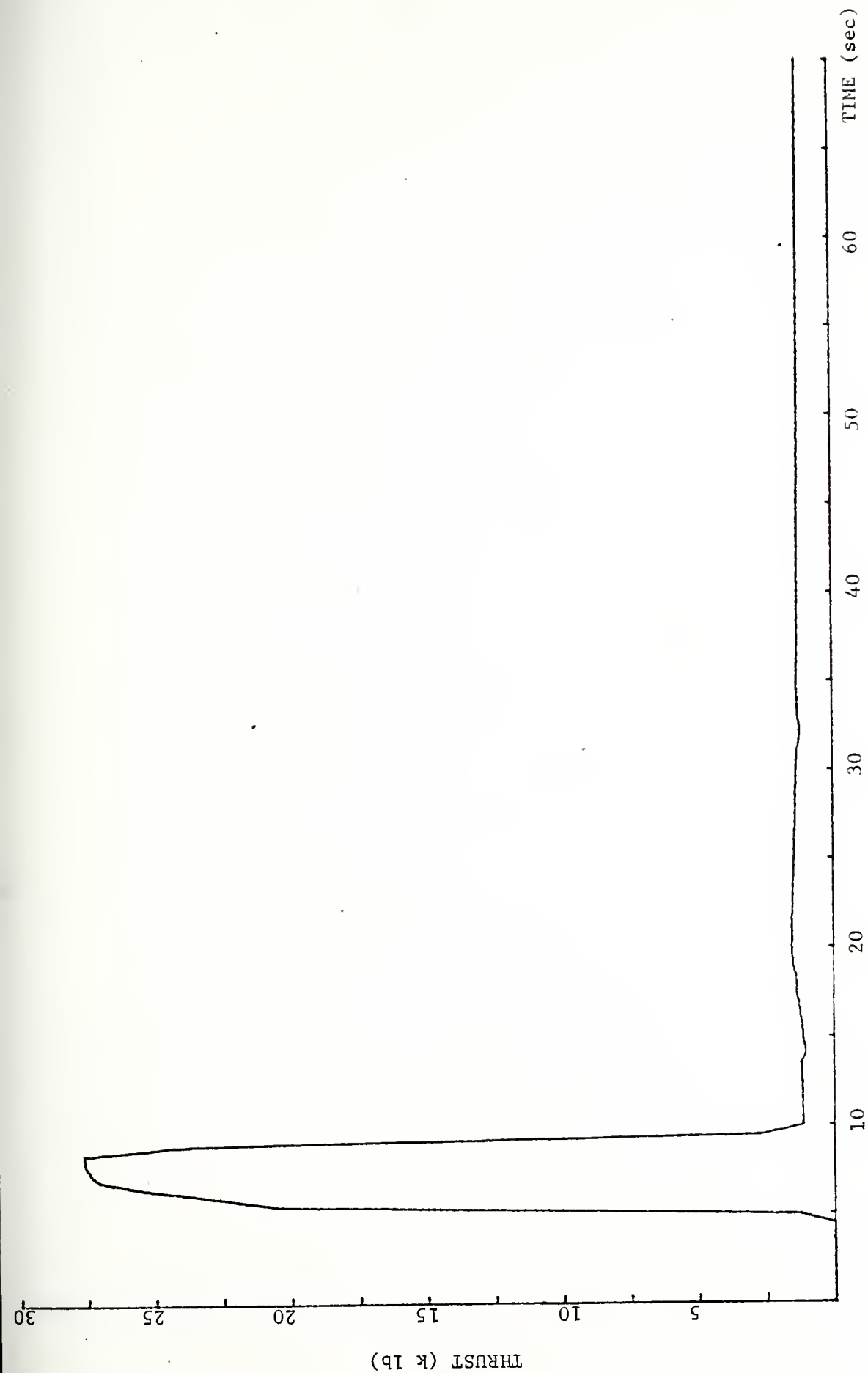
Rather than use RAMJET, the output disclosed that ENGINE had been used. The problem was in the logic statement, FLGRJ 0.0. The program was not making the desired switch to the simplified ramjet model. To remedy the problem, the data card FLGRJ(606) 1.0, was added to the input data deck. Another problem was discovered which occurred between 14.0 and 14.5 seconds. During that time period the missile angle of attack (ALPHA (330)) exceeded ten degrees. When ALPHA exceeds this angle the engine is turned off. With the engine off, no thrust is produced causing the forward velocity (VXTP(286)) to decrease. This limitation was removed from the program. After eliminating these problems a flight trajectory, Figure 7, was obtained which closely resembled the output of the mid-course baseline. A random sampling of the outputed variables were compared for exactness. The differences noted were due primarily to computer round-off error.

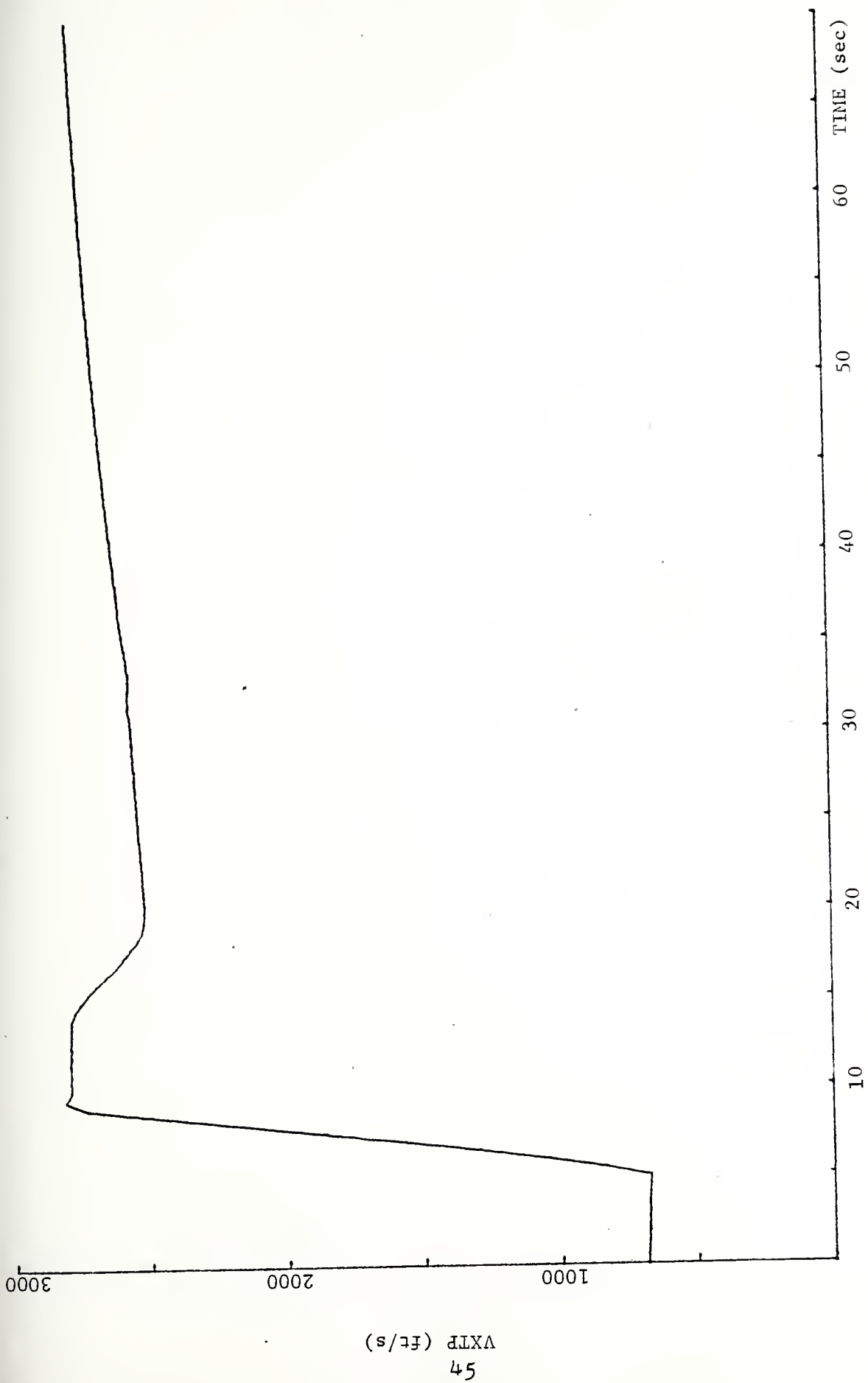
To further verify the accuracy of the two simulations, three parameter (TXBA(073), VXTP(286), and YTP(298)) were chosen as representative values for the runs. To obtain a feel for the run, these parameters are plotted in Figures 8, 9, 10. Additionally, random time samples are tabulated in Tables I, II, III.

The thrust (TXBA(073)) plot, Figure 8, can easily be divided into four mission phases. During the separation phase (0.0 - 4.5 seconds) the thrust is zero. This is expected since neither of the engines have ignited. This is followed by a rapid increase in the thrust. The maximum thrust, 27800 lbs, happens during the boost phase (4.5 - 9.5 seconds). At 9.5 seconds the boost motor stops and the transition phase occurs for the next 0.3 seconds. During this time the thrust decreases rapidly since neither engine is on. Once the engine port covers are

THRUST VS. TIME (MIDCOURSE)

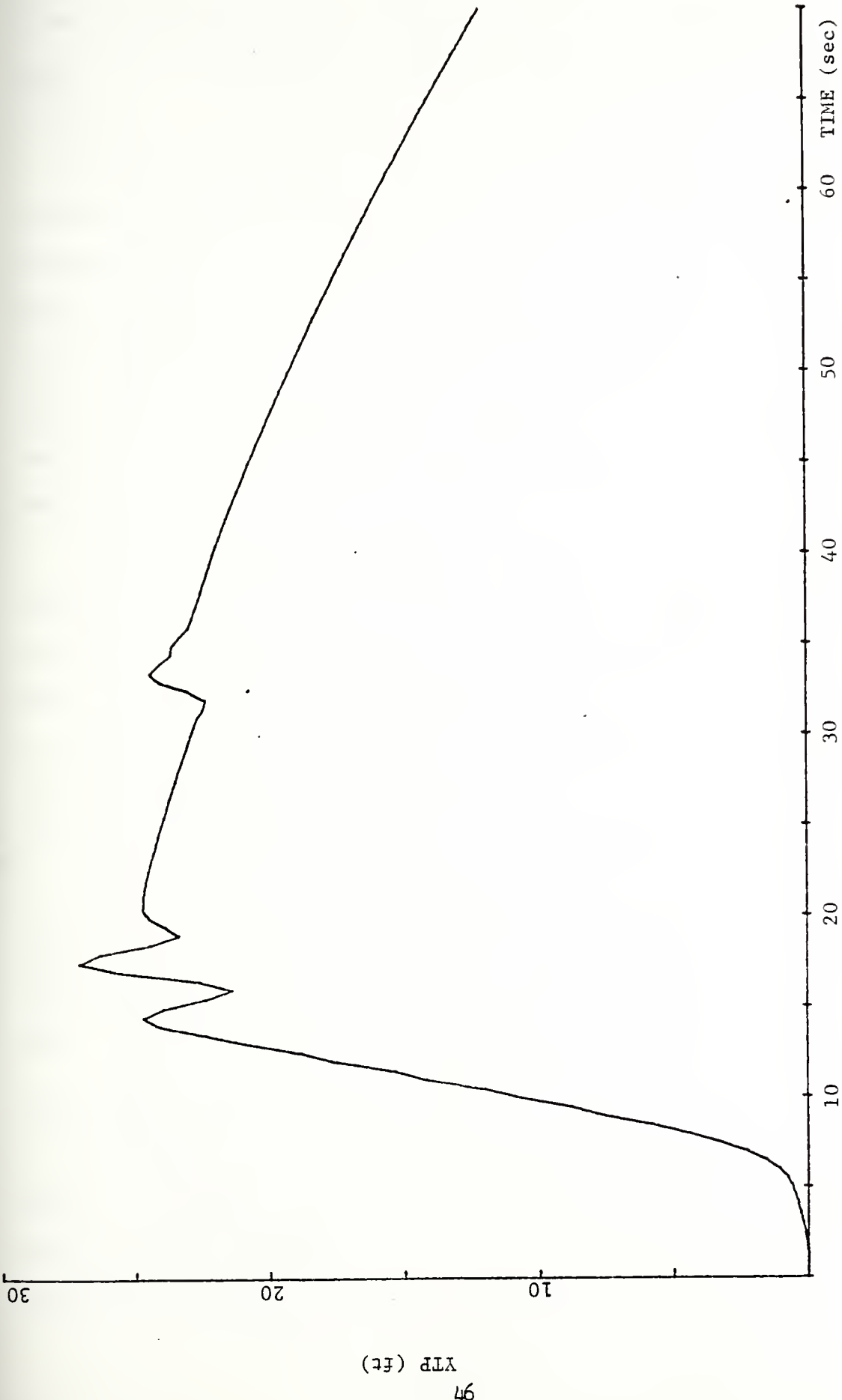
FIGURE 8





VXTP VS. TIME (MIDCOURSE)

FIGURE 9



YTP VS. TIME (MIDCOURSE)

FIGURE 10

clear, the ramjet sustainer ignites. This action initiates the midcourse phase (9.8 - 70 seconds) and the thrust remains relatively constant at 1200 lbs.

The missile forward velocity (VXTP(286)) can be directly related to the thrust. During separation no thrust is produced, therefore the velocity is zero. As the missile is boosted to Mach two, the velocity increases and approaches its maximum value (2800 ft/s) just prior to boost engine shut-down. As the sustainer engine port covers clear, the velocity decreases slightly. Once the ramjet engine ignites the velocity profile is displayed in Figure 9 and it closely resembles that for the midcourse baseline.

The last parameter, y-displacement (YTP(298)), was included to demonstrate the accuracy of the guidance system. Figure 10 shows that prior to attaining the cruise altitude, the missile wanders off track. Once the guidance system is activated, it makes the necessary corrections to return the missile to the planned flight path. Table III shows that the y-displacement corresponds to the baseline case and at the conclusion of the run the off-track error is down to 12.03 feet.

All the information obtained from the new run was checked against the midcourse baseline. The results showed that the two simulations are within acceptable limits. With this milestone completed, the program checkout could proceed to the terminal guidance flight.

B. TERMINAL GUIDANCE FLIGHT

To checkout the terminal guidance portion of the MOD6DF program, the initial conditions from the terminal baseline were used. This required changing about a dozen input cards in the midcourse input data deck. The first run attempted turned out successful. This was due to the

TIME	0.0	7.5	10.0	20.0	25.5	30.0	40.0	50.0	60.0	70.0
BASELINE	0.0	27395	1140.6	1403.0	1377.4	1281.0	1211.3	1178.0	1148.1	1124.8
NEW	0.0	27391	1138.9	1514.3	1418.4	1314.3	1249.9	1207.0	1175.5	1149.7

TXBA (073) COMPARISON (MIDCOURSE)

TABLE I

TIME	0.0	7.5	10.0	20.0	25.0	30.0	40.0	50.0	60.0	70.0
BASELINE	681.99	1858.3	2790.2	2606.7	2604.9	2613.4	2652.4	2705.0	2743.8	2771.5
NEW	681.99	1851.7	2786.3	2506.4	2525.9	2544.5	2605.0	2665.5	2710.8	2743.9

VXTP (286) COMPARISON (MIDCOURSE)

TABLE II

TIME	0.0	7.5	10.0	20.0	25.0	30.0	40.0	50.0	60.0	70.0
BASELINE	-5E-5	3.46	11.37	27.21	25.92	24.34	20.45	16.93	13.24	9.39
NEW	-5E-5	3.19	10.68	24.43	23.93	22.82	21.96	19.14	15.81	12.03

YTP (298) COMPARISON (MIDCOURSE)

TABLE III

TIME	0	5.0	10.0	12.6	15.0	17.6	20.0	22.6	25.0	TF
BASELINE	1141.9	1131.0	1077.8	1170.9	1309.4	1501.1	1730.0	2245.3	2565.8	2691.3
NEW	1141.6	1134.8	1099.4	1212.8	1359.3	1561.7	1800.0	2305.9	2584.3	3273.6

TXBA (073) COMPARISON (TERMINAL)

TABLE IV

TIME	0.0	5.0	10.0	12.6	15.0	17.6	20.0	22.6	25.0	TF
BASELINE	2750.0	2764.2	2654.6	2471.4	2225.7	1854.5	1391.4	1376.9	1413.1	1425.05
NEW	2750.0	2761.5	2631.5	2438.7	2192.1	1828.2	1389.3	1366.7	1406.1	1465.2

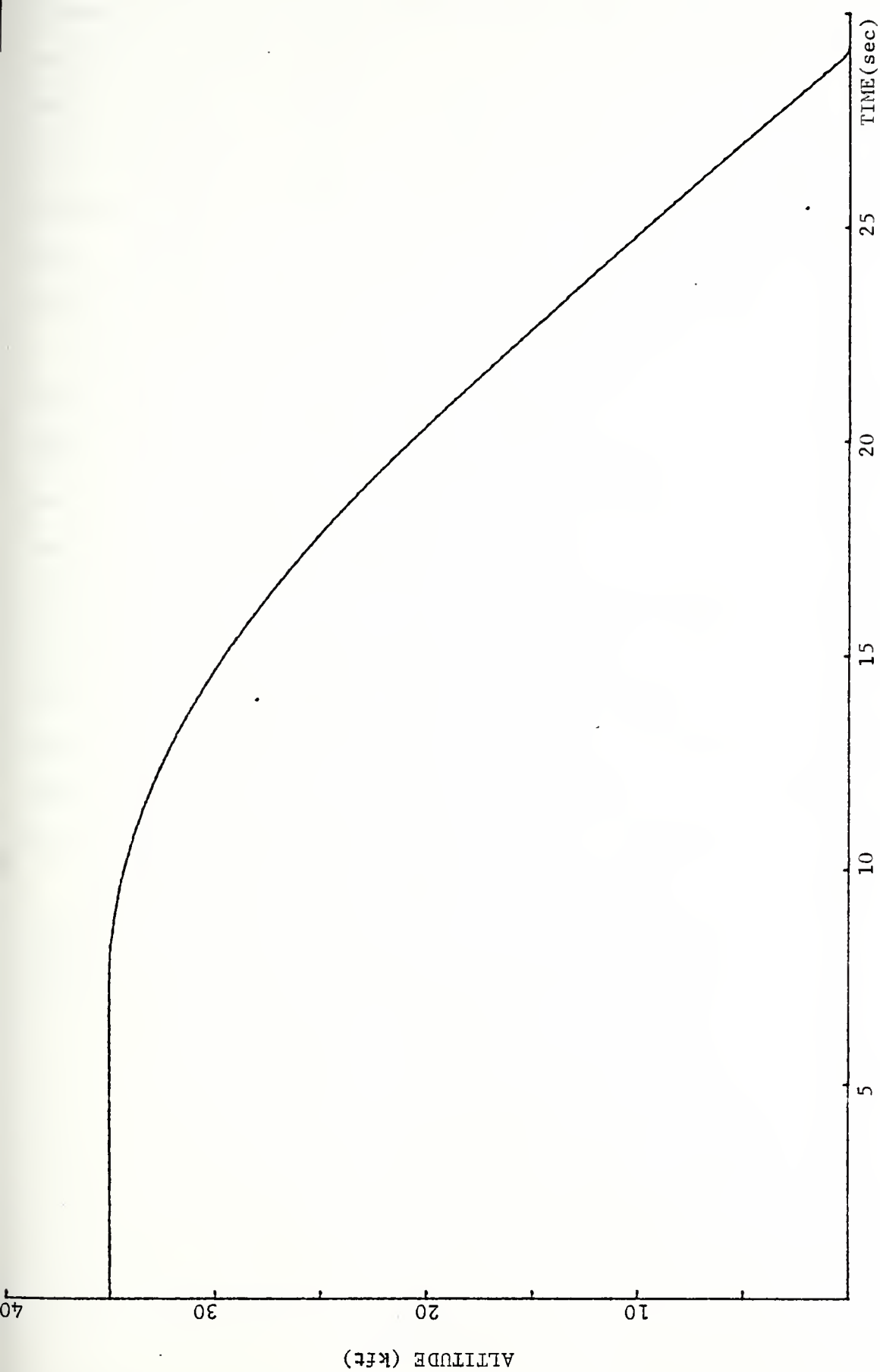
VXTP (286) COMPARISON (TERMINAL)

TABLE V

TIME	0.0	5.0	10.0	12.6	15.0	17.6	20.0	22.6	25.0	TF
BASELINE	0.0	0.0	- .4E-6	- .5E-6	.7E-6	.9E-6	.1E-5	.2E-6	.1E-6	.1E-6
NEW	0.0	- .0074	.019	-1.38	-.095	-.868	-1.77	.071	-.151	-.05

YTP (298) COMPARISON (TERMINAL)

TAVLE VI



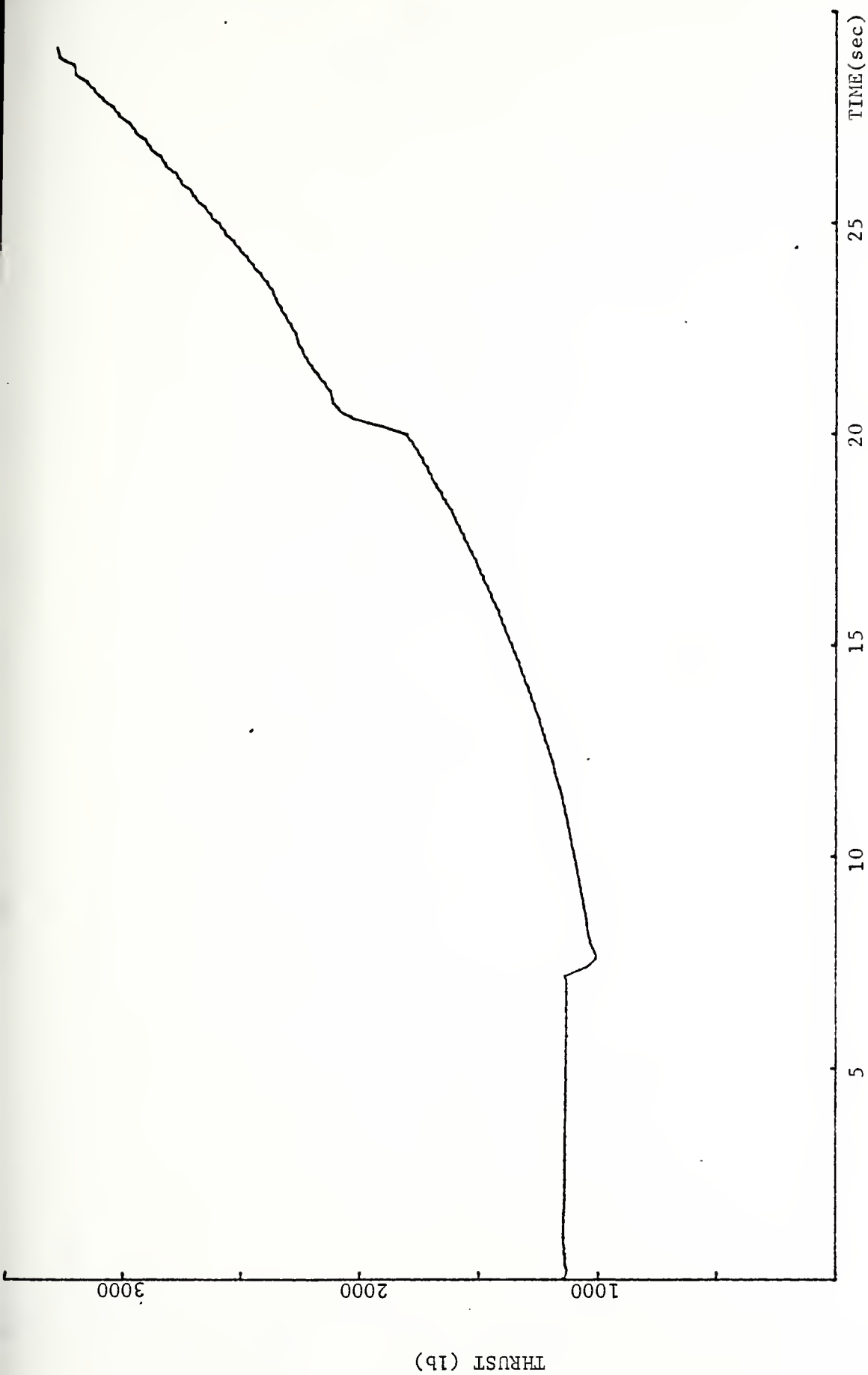
ALTITUDE VS. TIME (TERMINAL)

FIGURE 11

elimination of all the errors during the midcourse checkout. The resulting flight trajectory, Figure 11, is a very good illustration of the desired profile. To check the accuracy of the simulation, the same three parameters were chosen. Tables IV and V show that this run is almost identical to the terminal baseline. The difference is primarily due to computer round-off error. Figures 12 and 13 show the overall flight performance of the thrust and forward velocity, respectively.

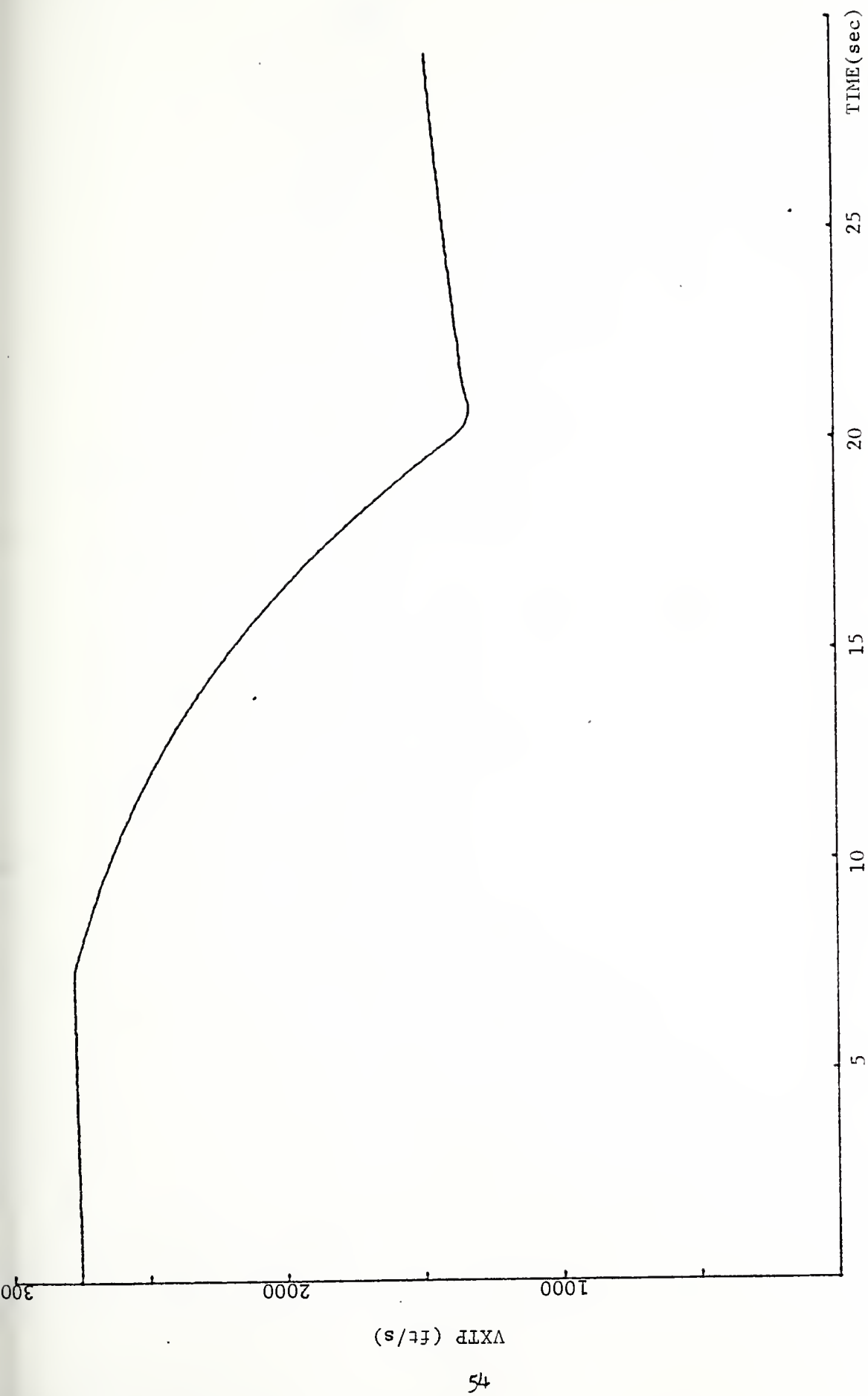
For this simulation, the y-displacement (YTP(298)) does not compare favorably with the terminal baseline. Figure 14 shows that between ten and twenty seconds in the flight, YTP develops rapid oscillations. By examining Table VI, it is noted that during this interval the two simulations differ the most. No reason for this discrepancy could be found. These oscillations do effect the flight by increasing the time of flight from 25.825 to 29.09 seconds and by increasing the miss distance. The imposed time constraints force this problem to be overlooked and to direct attention to the accuracy of the impact. Since the impact is within 0.5 feet of the target location, it is concluded that the terminal guidance simulation works properly.

With both segments of the MOD6DF computer program working correctly, thoughts could now turn towards the actual experimental runs. The results of these runs are discussed in the next section.



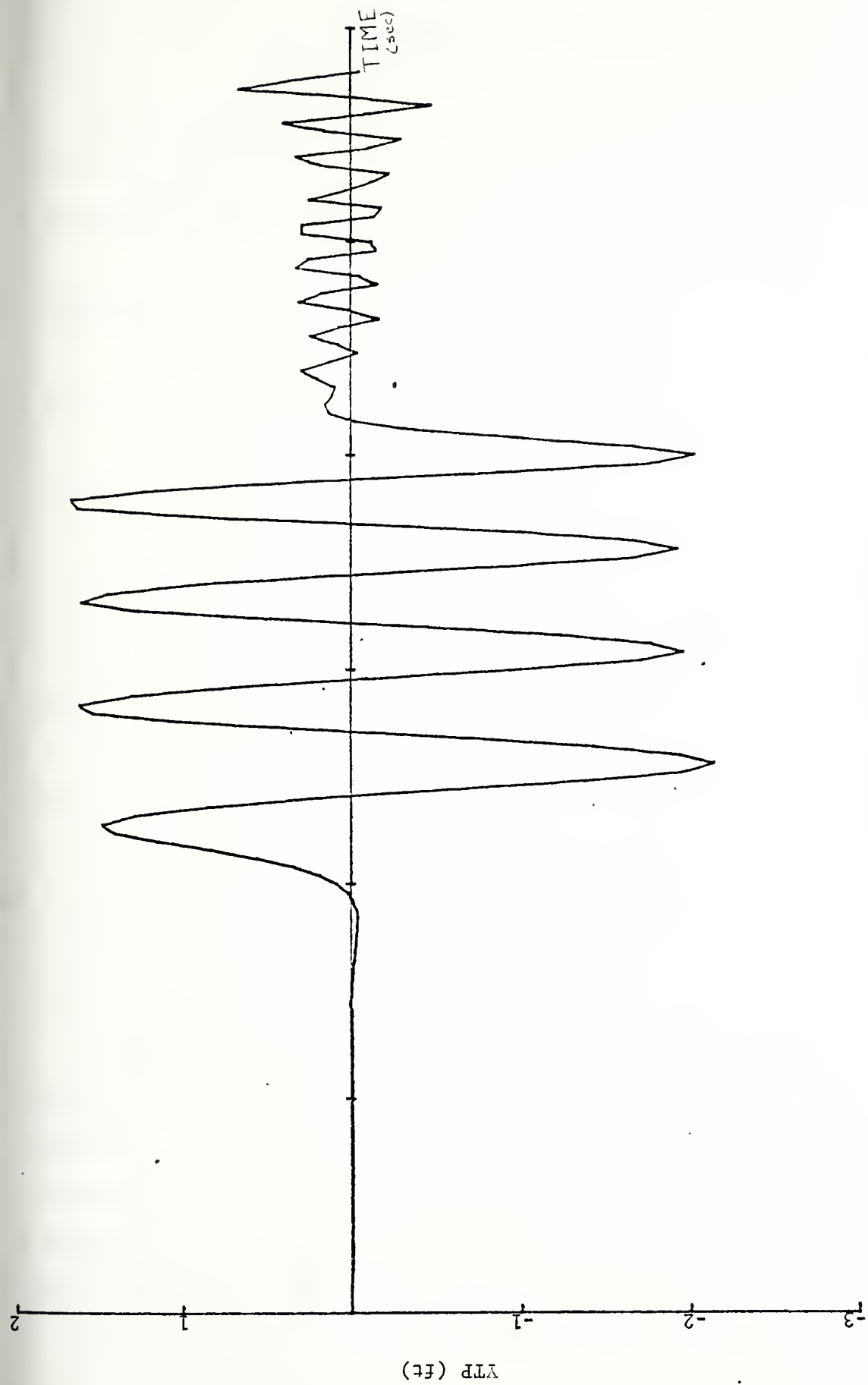
THRUST VS. TIME (TERMINAL)

FIGURE 12



VXTP VS. TIME (TERMINAL)

FIGURE 13



YTP VS. TIME (TERMINAL)

FIGURE 14

V. TERMINAL FLIGHT DISTURBANCES

With the checkout of the MOD6DF computer program completed, many ideas were discussed concerning alterations to the simulations. The three modifications decided upon were:

- * examine target miss distance when changes were made to the initial x-y-z conditions,
- * examine the terminal flight profile, range, and miss distance when the cruise altitude was reduced to approximate a sea skimming mode,
- * examine target miss distance when random noise is applied to the missile homing seeker.

To understand the changes and results, one must be familiar with the frame of reference. The origin of the reference frame travels from the launch platform to the target location at the cruise altitude. This is called the tangent plane reference system. Displacements in the x-direction (XTP(290)) are measured from the launch platform in the direction of the target. Y-displacements (YTP(298)) are measured left or right of the ideal flight path, in the tangent plane. Any vertical displacement (ZTP(306)) is measured normal to the ideal flight path, with down being the positive direction. Using this frame of reference the information in Table VII is easier to understand.

The first modifications demonstrate the effect of changing YTP. YTP was increased until a result was reached that was unsatisfactory. From the results in Table VIII, the maximum value of YTP was determined to be 1800 feet. This conclusion corresponds with the results in reference 13. Since the missile is symmetric, it was also concluded that moving YTP either right or left would give the same results.

The fifth run simulated a drop in the missile's altitude. ZTP was

RUN	XTP	YTP	ZTP
BASELINE	173552.5	0.0	0.0
1	173552.5	1000.0	0.0
2	173552.5	1500.0	0.0
3	173552.5	1800.0	0.0
4	173552.5	2000.0	0.0
5	173552.5	0.0	2000.0

INITIAL CONDITIONS

TABLE VII

RUN	XTP	YTP	ZTP	IMPACT TIME
BASELINE	234999.56	-.0506363	34999.34	28.825
1	234999.81	.1259258	34999.836	29.108
2	235000.19	.11479032	35000.32	29.124
3	235000.25	.15263242	35000.391	29.134
4	269802.0	2000.0	19705.00	35.0
5	234999.63	.45691	35000.426	30.365

IMPACT RESULTS

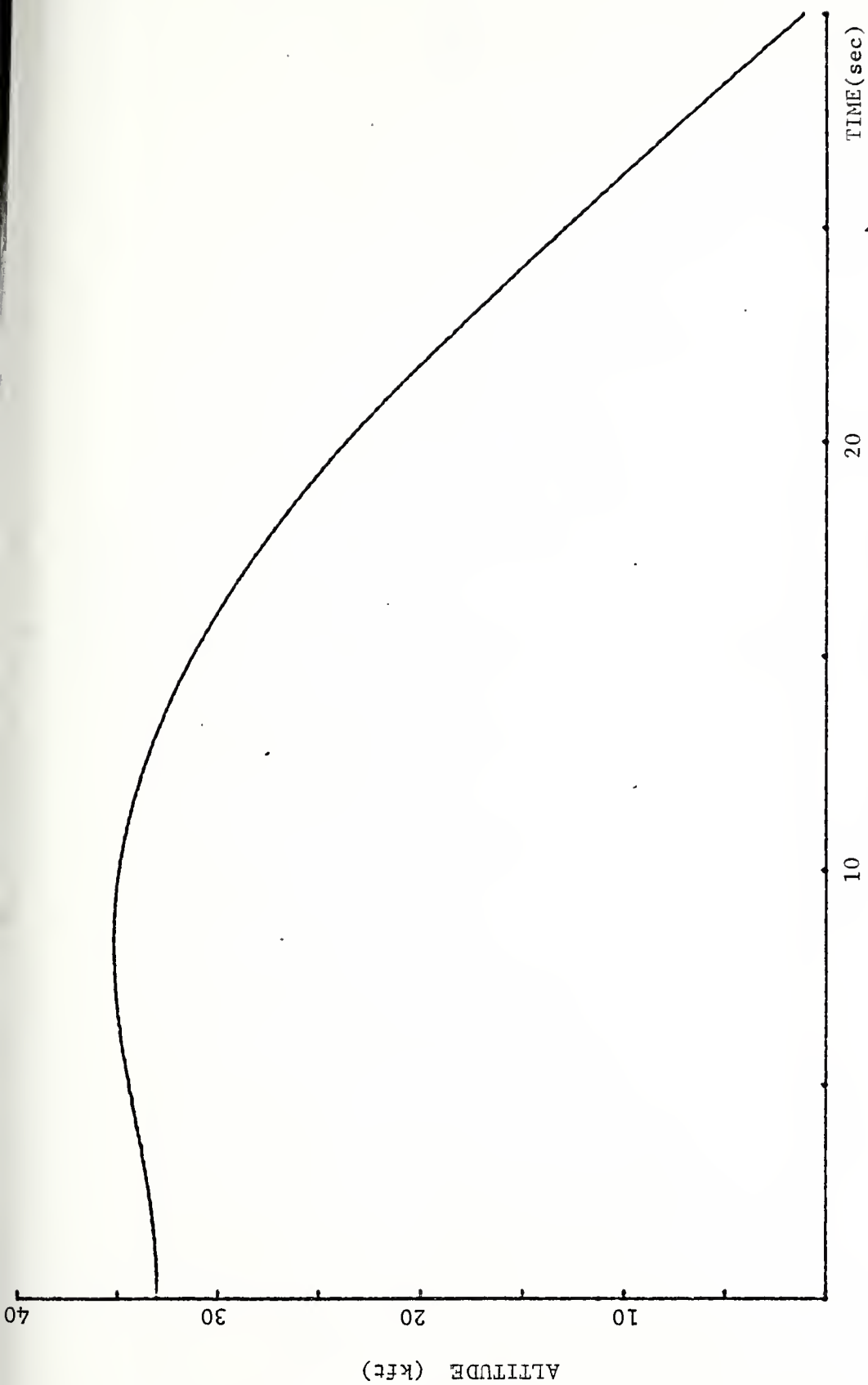
TABLE VIII

inputed as 2000.0 feet. As the simulation progressed, the missile developed the necessary commands to climb and regain the desired cruise altitude (Figure 15). It then commenced its terminal dive. The target was detected, acquired, and impact followed. This initial condition modification produced the largest miss distance (0.457 ft.).

Figure 16 was presented to show the relative locations of the miss distances for each change. Table VIII reveals that up until 1800 feet, changes to YTP created very little variation in the miss distance. It should be noted that the larger the displacement became, the greater the time until impact. The increase in time was necessary to allow the missile to acquire the target and then compute the required actuator commands to impact the target. This concluded the investigation of the effects of initial displacement error on target miss distance.

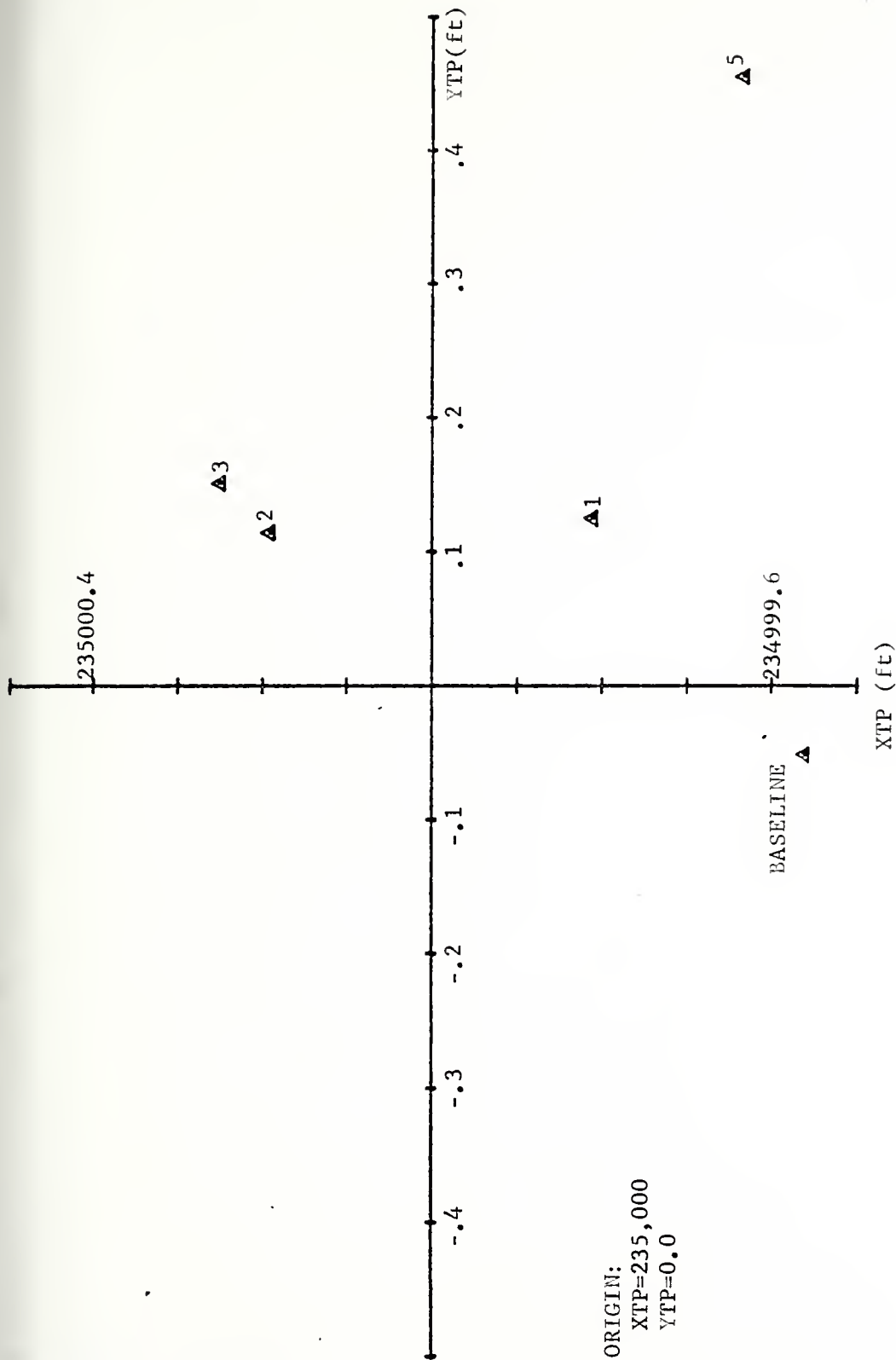
The idea behind changing the cruise altitude was to try and simulate a sea skimming profile. RAMJET contains tables that allow for four cruise altitudes. Of these four altitudes, the lowest (500 ft) was chosen even though it is high for a sea-skimmer. To run this simulation only two input data card changes were required. HO(414) and HREF(501) had to be set equal to the desired altitude. The resultant flight path is shown in Figure 17.

The simulation produced an error-free output, but at first glance the results appeared unacceptable. Still trying to compare miss distances, the downrange distance (XTP) was found to be 197916 feet. Comparing this to the terminal baseline (234999) resulted in an extreme error. It was then realized that the missile must expend more fuel at this altitude to attain the same speed. Therefore, the results could be correct. As further verification of the correctness of the run, it was assumed that the missile weight at impact should be fairly equal. The



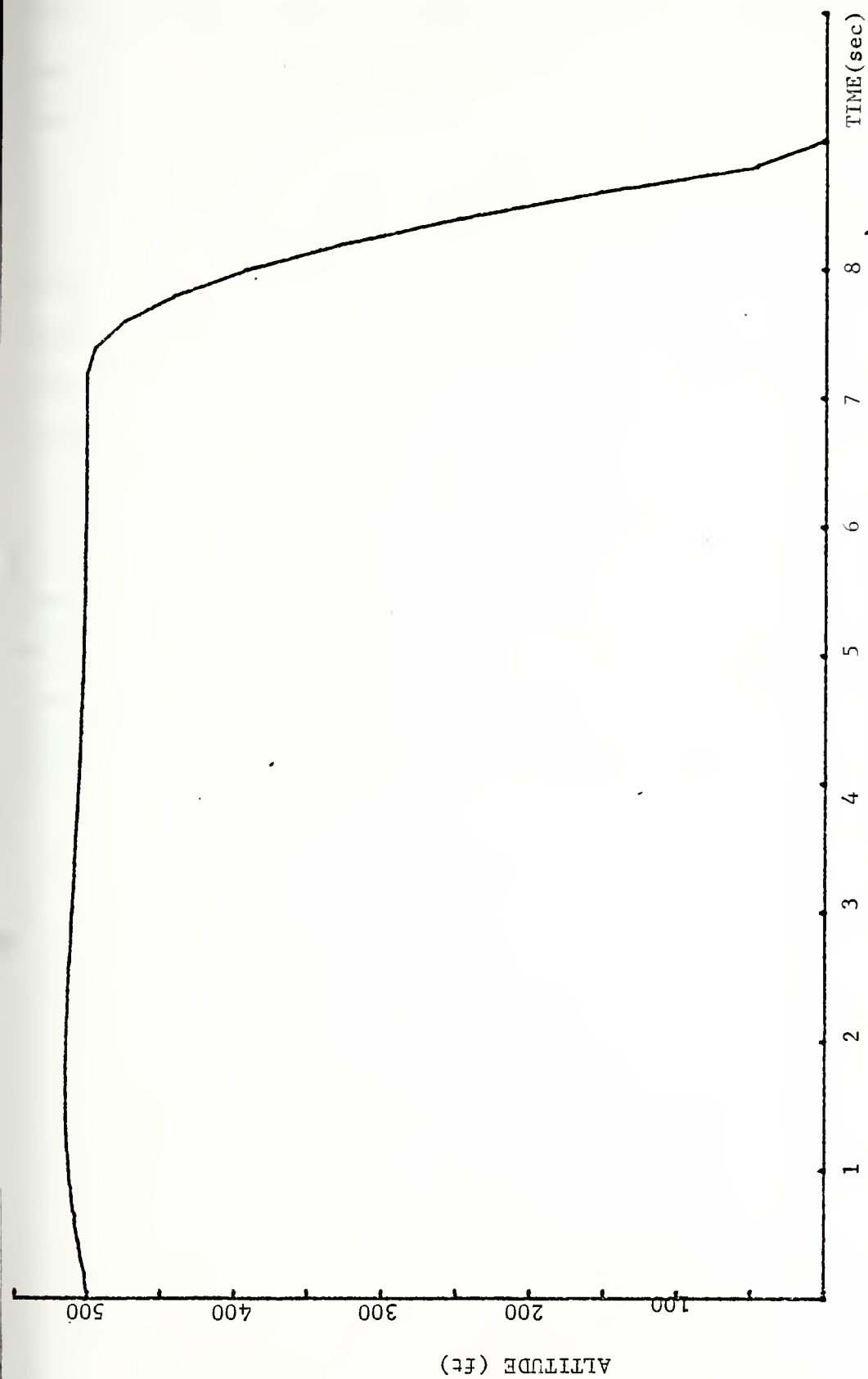
ALTITUDE VS. TIME (TERMINAL - ZTP)

FIGURE 15



IMPACT RESULTS

FIGURE 16



ALTITUDE VS. TIME (SEA-SKINNER)

FIGURE 17

simulation impact weight was 1126.34 lbs while the terminal baseline weighed 1113.12 lbs. Since no conflicting information could be found in reference 12, it was concluded that this simulation was correct.

The final flight disturbance investigated was the addition of random noise to the missile homing seeker. This simulation proved to be unsuccessful due to invalid input format. The information describing the process is brief and difficult to understand. Further information from NWC China Lake is required to be able to successfully run simulations with random noise.

With the completion of these modifications, many unanswered problems and questions still exist. The two most critical problems concern the input of random noise generators and the missile exceeding the angle of attack limitation when using ENGINE for cruise propulsion. However, the simulations did show that the MOD6DF program runs correctly at alternate cruise altitudes and with initial displacement errors.

VI. CONCLUSION

The MOD6DF computer program from NWC China Lake was converted to operate on the IBM-360 computer at the Naval Postgraduate School. The program would only function properly when using the simplified ramjet model. When this model was not used, the missile angle of attack exceeded the maximum limit of ten degrees. This error caused the ramjet engine to flame out.

Target impact errors for the terminal guidance problem were investigated when the initial displacement was modified. These modifications demonstrate the missile's accuracy when removed from the ideal flight path. The results also point out that if the target falls within the seeker search pattern, target impact will inevitably happen.

Random noise generation is possible with the MOD6DF program. However, more information discussing the parameters required to develop the noise is necessary. Additionally, sample noise inputs should be obtained that reflect the alteration to the noise subroutines.

This research involved the preliminary investigation of the MOD6DF program. The program has many possible areas, concerning guidance and control of tactical missiles, which could be developed for future study. These areas not only include the unresolved problems encountered during this research. Additionally, reference 14 and 15 contain many examples of possible advanced guidance concepts.

COMPUTER OUTPUT

65

TIME	THRUST	WIGHT	X(TP)	Y(TP)	Y(P)
	C	ALT	Y(TP)		ZP
	F	VELOCITY	Z(TP)		Z(TP)
	P	PITCH	ALPHA(DPG)	W/P	F(TA(DPG)
	ANYC	ANACE	DELTA C(DPG)	W/T	F(TA F(DPG)
	TALEF PARCIN	CYN P	YCW	F(TI	CASZV
0.49559950F-02	0.11416750E-04 0.25504540E-01 0.0 0.0 0.0	0.11525599E-04 0.36000000E-05 0.27500180E-04 0.27764110E-01 0.28155749E-01 0.16260527E-02	0.17356625E-06 0.0 0.45066515E-04 0.17630310E-01 0.8675882E-01 0.0	0.075018E-04 0.0 0.010721E-01 0.00507E-00 0.0 0.0	0.0 -0.4854611E-02 -0.2884454E-02 0.0 -0.0 -0.6045755E-02
0.20999952F-00	0.11325558E-04 0.52325558E-02 0.70484524E-02 0.77175721E-02 0.91046725E-04 0.0	0.11522553E-04 0.35111273E-05 0.27627713E-01 0.2762765E-01 0.28658757E-01 0.15259421E-02	0.17413030E-06 0.0 0.5167818E-03 -0.2766323E-01 0.2352231E-01 0.3388245E-01 -0.50249452E-04	0.0700001E-04 0.0 0.00124E-01 -0.00124E-01 0.00124E-01 -0.00124E-01 -0.00124E-01	0.0 -0.4854611E-02 -0.2884454E-02 0.0 -0.0 -0.6045755E-02
0.40499951E-00	0.11308408E-04 0.52325558E-02 0.70484524E-02 0.77175721E-02 0.91046725E-04 0.0	0.11522553E-04 0.35111273E-05 0.27627713E-01 0.2762765E-01 0.28658757E-01 0.15259421E-02	0.17413030E-06 0.0 0.5167818E-03 -0.2766323E-01 0.2352231E-01 0.3388245E-01 -0.50249452E-04	0.0700001E-04 0.0 0.00124E-01 -0.00124E-01 0.00124E-01 -0.00124E-01 -0.00124E-01	0.0 -0.4854611E-02 -0.2884454E-02 0.0 -0.0 -0.6045755E-02
0.60455984E-00	0.11308408E-04 0.52325558E-02 0.70484524E-02 0.77175721E-02 0.91046725E-04 0.0	0.11522553E-04 0.35111273E-05 0.27627713E-01 0.2762765E-01 0.28658757E-01 0.15259421E-02	0.17413030E-06 0.0 0.5167818E-03 -0.2766323E-01 0.2352231E-01 0.3388245E-01 -0.50249452E-04	0.0700001E-04 0.0 0.00124E-01 -0.00124E-01 0.00124E-01 -0.00124E-01 -0.00124E-01	0.0 -0.4854611E-02 -0.2884454E-02 0.0 -0.0 -0.6045755E-02
0.80455983F-00	0.11308408E-04 0.52325558E-02 0.70484524E-02 0.77175721E-02 0.91046725E-04 0.0	0.11522553E-04 0.35111273E-05 0.27627713E-01 0.2762765E-01 0.28658757E-01 0.15259421E-02	0.17413030E-06 0.0 0.5167818E-03 -0.2766323E-01 0.2352231E-01 0.3388245E-01 -0.50249452E-04	0.0700001E-04 0.0 0.00124E-01 -0.00124E-01 0.00124E-01 -0.00124E-01 -0.00124E-01	0.0 -0.4854611E-02 -0.2884454E-02 0.0 -0.0 -0.6045755E-02

COMPUTER PROGRAM

THIS IS THE MAIN MCD6DF PROGRAM

```

CGMAGN C(3415), TFMP5(1500)
EQUIVALENCE (C(3315),N)
EQUIVALENCE (C(2911),HMIN)
EQUIVALENCE (C(2912),HMAX)
EQUIVALENCE (C(2913),CFR(1))
EQUIVALENCE (C(3014),VAR(1))
EQUIVALENCE (C(3115),EL(1))
EQUIVALENCE (C(3215),EL(1))
EQUIVALENCE (C(3316),IPL(1))
EQUIVALENCE (C( 932),T)
EQUIVALENCE (C(2904),KSTEP), (C(2905),STEP)
EQUIVALENCE (C(2907),LSTEP)
DIMENSION IPL(100)
DIMENSION DER(101)
DIMENSION VAR(101)
DIMENSION VEL(100)
DIMENSION EU(100)
N=M
N=M
CALL ZERO
CALL CINSTEP
CALL SUELI
CALL AUXI
CALL SUELI2,N
DO 60 I=2,N
  J=IPL(I-1)
  EU(I-1)=C(J+1)
  EU(I-1)=C(J+2)
  VAR(I)=C(J+3)
  DER(I)=C(J)
  VAR(I)=I
  CALL AUXSUB
  CALL AMK
  DO 50 I=2,N
    J=IPL(I-1)
    C(J+3)=VAP(I)
    T=VAR(I)
    CALL SUELI3
  1009 IF (KSTEP.EQ. 1) GC TC 1007
    IF (KSTEP-1) 70,1007,7C
    CALL PRECET
    GO TC (1000,1001,1002,1003,1004,1005,1006,1007,1008,1009,1010),

```

(C(2905),STEP)

```

100C CALL ZERO
1001 CALL CINSTEP
1002 CALL SUELI
1003 CALL AUXI
1004 CALL SUELI2,N
1005 DO 60 I=2,N
      J=IPL(I-1)
      EU(I-1)=C(J+1)
      EU(I-1)=C(J+2)
      VAR(I)=C(J+3)
      DER(I)=C(J)
      VAR(I)=I
      CALL AUXSUB
      CALL AMK
      DO 50 I=2,N
        J=IPL(I-1)
        C(J+3)=VAP(I)
        T=VAR(I)
        CALL SUELI3
      1009 IF (KSTEP.EQ. 1) GC TC 1007
        IF (KSTEP-1) 70,1007,7C
        CALL PRECET
        GO TC (1000,1001,1002,1003,1004,1005,1006,1007,1008,1009,1010),

```

MCD60010
MCD60020
MCD60030
MCD60040
MCD60050
MCD60060
MCD60070
MCD60100
MCD60110
MCD60120
MCD60130
MCD60140
MCD60150
MCD60160
MCD60170
MCD60200
MCD60210

MCD60220
MCD60230
MCD60240
MCD60250
MCD60260
MCD60270
MCD60300
MCD60310
MCD60320
MCD60330
MCD60340
MCD60350
MCD60360
MCD60370
MCD60400
MCD60410
MCD60420
MCD60430
MCD60440
MCD60450
MCD60460
MCD60470
MCD60500
MCD60510
MCD60520

MCD60530
 MCD60540
 MCD60545
 MCD60550

1 LSTEP
 1010 CALL EXIT
 STOP
 END

SUBROUTINE ZERO

COMMON C(3415)
 DO 1 KLEAR = 1, 4915
 1 C(KLEAR) = 0.0
 RETURN
 END

SUBROUTINE QINPT1

BASIC INFLT SUBROUTINE CINPT1
 SUBROUTINE QINPT1 IS THE BASIC INPUT SUBROUTINE

DIMENSION LISTNO(50), VALUE(50)
 DIMENSION SUBNO(99), IR(2), VR(2)
 DIMENSION RNDMNO(50)
 DIMENSION ALPHA(4), CNAME1(50), CNAME2(50), CNAME3(50), OUTNC(50)
 DIMENSION MCCNO(99)
 DIMENSION STATNO(100)
 COMMON C(3415)

REAL MCDAC
 INTEGER CUTNO
 INTEGER RNDMNO
 INTEGER STATNO
 EQUIVALENCE (C(2442), LOSTAT)
 EQUIVALENCE (C(2200), STATNO(1)), (C(2441), NOSTAT)
 EQUIVALENCE (C(2801), NOSUB), (C(2802), SUBNO(1)), (C(2663), IR(1)),
 1 (C(2664), VR(1))
 EQUIVALENCE (C(2701), NCMOD)
 EQUIVALENCE (C(2661), NCOUT)
 EQUIVALENCE (C(2448), NORNDM)
 EQUIVALENCE (C(2300), NOLIST), (C(2301), LISTNO(1)), (C(2351),
 1 VALUE(1))
 EQUIVALENCE (C(2500), CUTNO(1))
 EQUIVALENCE (C(2550), CNAME1(1))
 EQUIVALENCE (C(2600), CNAME2(1))
 EQUIVALENCE (C(2150), CNAME3(1))
 EQUIVALENCE (C(2450), RNDMNO(1))
 EQUIVALENCE (C(2702), MCDNO(1))
 JAR = C

CINP0030
 CINP001C
 CINP0020

CINP0230
 CINP0240
 CINP0250

CINP030C
 CINP0040
 CINP031C
 CINP032C
 CINP0330
 CINP034C
 CINP005C

CINP011C
 CINP012C
 CINP0130

CINP0350


```

1 READ(5,2) IR(1),ALPHA(1),ALPHA(2),ALPHA(3),ALPHA(4),IR(2),VR(1),
1 VR(2)
WRITE(6,2) IR(1),ALPHA(1),ALPHA(2),ALPHA(3),ALPHA(4),IR(2),VR(1),
1 VR(2)
2 FORMAT(I2,2X,4A4,I5,5X,2E15.8)
IF (IR(1)) .NE. 0) GO TO 7
IF (IR(1)) 7,18,7
18 IR = IR(2)
IR = IR(1) - 1.
IR = VR(2)
IF (IR(1)) 11,15,14
14 DO 8 I = 1,12
8 READ (I,10) J,X
15 DO 9 I = 1,13
9 READ (I,2) J,X
C(J) = X
10 FORMAT (I5,E15.9)
11 JAR TC 1
11 GO TC 1
7 IF (IR(1)) .NE. 1) GO TO 3
7 IF (IR(1)) - 1) 3,19,3
19 NOSUB = NCSUB + 1
SUBNC(NCSUB) = IR(2)
GO TC 1
3 IF (IR(1)) .NE. 2) GO TC 4
3 IF (IR(1)) - 2) 4,20,4
20 NOMOD = NCMOD + 1
MODNC(NCMOD) = IR(2)
GO TC 1
4 IF (IR(1)) .NE. 3) GO TO 5
4 IF (IR(1)) - 3) 5,21,5
21 L = IR(2)
C(L) = VR(1)
IF (JAR .EQ. 1) WRITE (I1,10)L,VR(1)
IF (JAR - 1) 23,22,23
22 WRITE (I1,10)L,VR(1)
23 IF (VR(2)) .EQ. 0) GO TO 1
24 IF (VR(2)) 24,1,24
24 NOLIST = NCLIST + 1
LISTAC(NCLIST) = L
VALUE(NCLIST) = VR(1)
GO TC 1
5 IF (IR(1)) .NE. 4) GO TO 6
5 IF (IR(1)) - 4) 6,25,6
25 NOCUT = NCLUT + 1
CNAME1(NCCLT) = ALPHA(2)

```

```

C INP0410
C INP0420
C INP0430
C INP0440
C INP0450
C INP0460
C INP0470
C INP0500
C INP0510
C INP0520
C INP0530
C INP0540
C INP0550
C INP0560
C INP0570
C INP0600
C INP0610
C INP0620
C INP0630
C INP0640
C INP0650
C INP0660
C INP0670
C INP0700
C INP0710
C INP0720
C INP0730
C INP0740
C INP0750
C INP0760
C INP0770
C INP1000
C INP1010
C INP1020
C INP1030
C INP1040
C INP1050
C INP1060
C INP1070
C INP1100
C INP1120

```


6 JBL0200C
 7 JBL0210C
 8 JBL0220C
 9 JBL0230C
 10 JBL0240C
 11 JBL0250C
 12 JBL0260C
 13 JBL0270C
 14 JBL0300C
 15 JBL0310C
 16 JBL0320C

17 JBL0910
 18 JBL0050
 19 JBL0020C
 20 JBL0030C
 21 JBL0060C
 22 JBL0070C
 23 JBL0100C
 24 JBL0110C
 25 JBL0120C
 26 JBL0130C
 27 JBL0140C
 28 JBL0150C
 29 JBL0160C
 30 JBL0170C
 31 JBL0200C
 32 JBL0210C
 33 JBL0220C
 34 JBL0230C
 35 JBL0240C
 36 JBL0250C
 37 JBL0260C
 38 JBL0270C
 39 JBL0300C
 40 JBL0310C
 41 JBL0320C

42 JBL0010
 43 JBL0050C
 44 JBL0020C

6 GO TC 1
 7 CALL RNCM1
 8 GO TC 1
 9 CALL ALXA1
 10 GO TC 1
 11 CALL ALXB1
 12 GO TC 1
 13 CALL ALXC1
 14 CONTINUE
 15 RETURN
 16 END

SUBROUTINE SUBL2
 DIMENSION SUBNO(99)
 COMMON C(3415)
 EQUIVALENCE (C(2801),NCSLB)
 EQUIVALENCE (C(2802),SUBNO(1))
 DO 1 I = 1, NCSub
 J = SUBNO(I)
 GO TO (1, 2, 3, 4, 5, 6, 7, 8, 9), J
 2 CALL INFT2
 3 GO TC 1
 4 CALL CLFT2
 5 GO TC 1
 6 CALL STGE2
 7 GO TC 1
 8 CALL CNTR2
 9 GO TC 1
 10 CALL RNCM2
 11 GO TO 1
 12 CALL ALXA2
 13 GO TC 1
 14 CALL ALXB2
 15 GO TC 1
 16 CALL ALXC2
 17 CONTINUE
 18 RETURN
 19 END

SUBROUTINE SUBL3
 DIMENSION SUBNO(99)
 COMMON C(3415)

C
 C
 C
 C

C
 C
 C
 C


```

EQUIVALENCE (C(2801),NOSUB)
EQUIVALENCE (C(2802),SUENO(1))
DO 1 I = 1, NOSUB
J = SUENO(I)
GO TC (1, 2, 3, 4, 5, 6, 7, 8, 9), J
2 CALL INPT3
3 GO TC 1
4 CALL CUPT3
5 GO TC 1
6 CALL STGE3
7 GO TC 1
8 CALL CNTR3
9 GO TC 1
10 CALL RNDM3
11 GO TC 1
12 CALL ALXA3
13 GO TC 1
14 CALL ALXB3
15 GO TC 1
16 CALL ALXC3
17 CONTINUE
18 RETURN
19 END

```

C C C C

```

SUBROUTINE AUXI
DIMENSION MCDNO(99)
COMMON C(3415)
REAL MCDNO
EQUIVALENCE (C(2701), NCMOD)
EQUIVALENCE (C(2702), MCDNO(1))
EQUIVALENCE (C(2659), CNCE)
EQUIVALENCE (C(3315), N)
N = 1
CNCE = 0
DO 1 I = 1, NCMOD
L = MCDNO(I)
GO TC (1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37), L
1 CALL AL1
2 GO TC 1
3 CALL AL2
4 GO TC 1
5 CALL AL3
6 GO TC 1
7 CALL AL4
8 GO TC 1
9 GO TC 1
10 GO TC 1
11 GO TC 1
12 GO TC 1
13 GO TC 1
14 GO TC 1
15 GO TC 1
16 GO TC 1
17 GO TC 1
18 GO TC 1
19 GO TC 1
20 GO TC 1
21 GO TC 1
22 GO TC 1
23 GO TC 1
24 GO TC 1
25 GO TC 1
26 GO TC 1
27 GO TC 1
28 GO TC 1
29 GO TC 1
30 GO TC 1
31 GO TC 1
32 GO TC 1
33 GO TC 1
34 GO TC 1
35 GO TC 1
36 GO TC 1
37 GO TC 1

```

SURL0030
SURL006C
SURL0070
SURL0100
SURL011C
SURL0120
SURL013C
SURL0140
SURL015C
SURL016C
SURL0170
SURL020C
SURL021C
SURL022C
SURL023C
SURL024C
SURL0250
SURL026C
SURL027C
SURL030C
SURL0310
SURL032C

ALXI001C
ALXI006C
ALXI002C
ALXI0070
ALXI003C
ALXI005C
ALXI010C
ALXI011C
ALXI012C
ALXI013C
ALXI0140
ALXI0150
ALXI016C
ALXI017C
ALXI020C
ALXI021C
ALXI022C

5 CALL A4I
 6 GO TC 1I
 7 CALL A5I
 8 GO TC 1I
 9 CALL C1I
 10 GO TC 1I
 11 CALL C2I
 12 GO TC 1I
 13 CALL C3I
 14 GO TC 1I
 15 CALL C4I
 16 GO TC 1I
 17 CALL C5I
 18 GO TC 1I
 19 CALL C6I
 20 GO TC 1I
 21 CALL C7I
 22 GO TC 1I
 23 CALL C8I
 24 GO TC 1I
 25 CALL C9I
 26 GO TC 1I
 27 CALL C10I
 28 GO TC 1I

ALXI023C
 ALXI0240
 ALXI025C
 ALXI026C
 ALXI027C
 ALXI028C
 ALXI029C
 ALXI030C
 ALXI031C
 ALXI032C
 ALXI033C
 ALXI034C
 ALXI035C
 ALXI036C
 ALXI037C
 ALXI038C
 ALXI039C
 ALXI040C
 ALXI041C
 ALXI042C
 ALXI043C
 ALXI044C
 ALXI045C
 ALXI046C
 ALXI047C
 ALXI048C
 ALXI049C
 ALXI050C
 ALXI051C
 ALXI052C
 ALXI053C
 ALXI054C
 ALXI055C
 ALXI056C
 ALXI057C
 ALXI058C
 ALXI059C
 ALXI060C
 ALXI061C
 ALXI062C
 ALXI063C
 ALXI064C
 ALXI065C
 ALXI066C
 ALXI067C
 ALXI068C
 ALXI069C
 ALXI070C
 ALXI071C
 ALXI072C
 ALXI073C
 ALXI074C
 ALXI075C
 ALXI076C
 ALXI077C
 ALXI078C
 ALXI079C
 ALXI080C
 ALXI081C
 ALXI082C

ALXI1103C
ALXI1104C
ALXI1105C
ALXI1106C
ALXI1107C
ALXI1110C
ALXI1112C
ALXI1113C
ALXI1114C
ALXI1115C
ALXI1116C
ALXI1117C
ALXI1120C
ALXI1121C
ALXI1122C
ALXI1123C
ALXI1124C
ALXI1125C
ALXI1126C

ALXS001C
ALXS0120
ALXS0123C
ALXS0140C
ALXS015C
ALXS002C
ALXS0160
ALXS003C

ALXS010C
ALXS011C
ALXS0170C
ALXS020C
ALXS0210
ALXS0220C
ALXS023C
ALXS024C
ALXS025C
ALXS026C
ALXS0270
ALXS029C

25 CALL S2I
30 GO TC S2I
31 GO TC S4I
32 GO TC S5I
33 GO TC S6I
34 GO TC S7I
35 GO TC S8I
36 GO TC S9I
37 GO TC S10I
1 RETURN
2 END

SUBROUTINE AUXSUB

DIMENSION DER(101)
DIMENSION VAR(101)
DIMENSION IPL(100)
DIMENSION MCDNO(99)
COMMON C(3415)
REAL MCDNC
EQUIVALENCE (C(2701), NOMOD)
EQUIVALENCE (C(2702), MCDNO(1))
EQUIVALENCE (C(3316), IPL(1))
EQUIVALENCE (C(3014), VAR(1))
EQUIVALENCE (C(2913), DER(1))
EQUIVALENCE (C(932), T)
EQUIVALENCE (C(3315), N)
DO 5C I = 2, N
J = IFL(I-1)
C(J+3) = VAR(I)
T = VAR(I)
DO 1 I = 1, NCMC
L = MCDNC(I)
GO TC (1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,
123,24,25,26,27,28,29,30,31,32,33,34,35,36,37),L
2 CALL AI
3 GO TC 1

C C C C

3100
 3200
 3300
 3400
 3500
 3600
 3700
 4000
 4100
 4200
 4300
 4400
 4500
 4600
 4700
 5000
 5100
 5200
 5300
 5400
 5500
 5600
 5700
 6000
 6100
 6200
 6300
 6400
 6500
 6600
 6700
 7000
 7100
 7200
 7300
 7400
 7500
 7600
 7700
 1000
 1010
 1020
 1030
 1040
 1050
 1060
 1070
 1100

32
 41
 53
 64
 75
 81
 91
 101
 114
 125
 136
 147
 158
 169
 171
 181
 193
 204
 215
 221
 231
 241
 253
 264


```

27 CALL G6
28 GO TO 1
29 CALL S1
30 GO TC S1
31 CALL S2
32 GO TC S2
33 CALL S3
34 GO TC S3
35 CALL S4
36 GO TC S4
37 CALL S5
38 GO TC S5
39 CALL S6
40 GO TC S6
41 CALL S7
42 GO TC S7
43 CALL S8
44 GO TC S8
45 GO TO 1
46 CALL S9
47 GO TC S9
48 CALL S10
49 GO TC S10
50 CONTINUE
51 DO 60 I = 2, N
52 J = IFL(I-1)
53 DER(I) = (J)
54 RETURN
55 END

```

SUERCUTINE AMRK

DOUBLE PRECISION VERSION. SAVES FUNCTION VALUES AND THE INDEPENDANT VARIABLE IN FULL D.P.

```

DIMENSION V(1), D(1), EU(1), FL(1)
COMMON P1, P2, P3, P4, C2, C3, C4
COMMON T(I5C0), TME, DELT
COMMON T(3014), V(11)
COMMON C(2913), C(1)
COMMON C(3215), EU(1)
COMMON C(3115), FL(1)
COMMON C(2655), CNCE
COMMON C(3315), CNI
COMMON C(2912), HMAX
COMMON C(2911), TMIN
COMMON C(11, N1)
C(1) = 0
C(2) = 0
C(3) = 0
C(4) = 0
C(5) = 0
C(6) = 0
C(7) = 0
C(8) = 0
C(9) = 0
C(10) = 0
C(11) = 0
C(12) = 0
C(13) = 0
C(14) = 0
C(15) = 0
C(16) = 0
C(17) = 0
C(18) = 0
C(19) = 0
C(20) = 0
C(21) = 0
C(22) = 0
C(23) = 0
C(24) = 0
C(25) = 0
C(26) = 0
C(27) = 0
C(28) = 0
C(29) = 0
C(30) = 0
C(31) = 0
C(32) = 0
C(33) = 0
C(34) = 0
C(35) = 0
C(36) = 0
C(37) = 0
C(38) = 0
C(39) = 0
C(40) = 0
C(41) = 0
C(42) = 0
C(43) = 0
C(44) = 0
C(45) = 0
C(46) = 0
C(47) = 0
C(48) = 0
C(49) = 0
C(50) = 0
C(51) = 0
C(52) = 0
C(53) = 0
C(54) = 0
C(55) = 0
C(56) = 0
C(57) = 0
C(58) = 0
C(59) = 0
C(60) = 0
C(61) = 0
C(62) = 0
C(63) = 0
C(64) = 0
C(65) = 0
C(66) = 0
C(67) = 0
C(68) = 0
C(69) = 0
C(70) = 0
C(71) = 0
C(72) = 0
C(73) = 0
C(74) = 0
C(75) = 0
C(76) = 0
C(77) = 0
C(78) = 0
C(79) = 0
C(80) = 0
C(81) = 0
C(82) = 0
C(83) = 0
C(84) = 0
C(85) = 0
C(86) = 0
C(87) = 0
C(88) = 0
C(89) = 0
C(90) = 0
C(91) = 0
C(92) = 0
C(93) = 0
C(94) = 0
C(95) = 0
C(96) = 0
C(97) = 0
C(98) = 0
C(99) = 0
C(100) = 0
C(101) = 0
C(102) = 0
C(103) = 0
C(104) = 0
C(105) = 0
C(106) = 0
C(107) = 0
C(108) = 0
C(109) = 0
C(110) = 0
C(111) = 0
C(112) = 0
C(113) = 0
C(114) = 0
C(115) = 0
C(116) = 0
C(117) = 0
C(118) = 0
C(119) = 0
C(120) = 0
C(121) = 0
C(122) = 0
C(123) = 0
C(124) = 0
C(125) = 0
C(126) = 0
C(127) = 0
C(128) = 0
C(129) = 0
C(130) = 0
C(131) = 0
C(132) = 0
C(133) = 0
C(134) = 0
C(135) = 0
C(136) = 0
C(137) = 0
C(138) = 0
C(139) = 0
C(140) = 0
C(141) = 0
C(142) = 0
C(143) = 0
C(144) = 0
C(145) = 0
C(146) = 0
C(147) = 0
C(148) = 0
C(149) = 0
C(150) = 0
C(151) = 0
C(152) = 0
C(153) = 0
C(154) = 0
C(155) = 0
C(156) = 0
C(157) = 0
C(158) = 0
C(159) = 0
C(160) = 0
C(161) = 0
C(162) = 0
C(163) = 0
C(164) = 0
C(165) = 0
C(166) = 0
C(167) = 0
C(168) = 0
C(169) = 0
C(170) = 0
C(171) = 0
C(172) = 0
C(173) = 0
C(174) = 0
C(175) = 0
C(176) = 0
C(177) = 0
C(178) = 0
C(179) = 0
C(180) = 0
C(181) = 0
C(182) = 0
C(183) = 0
C(184) = 0
C(185) = 0
C(186) = 0
C(187) = 0
C(188) = 0
C(189) = 0
C(190) = 0
C(191) = 0
C(192) = 0
C(193) = 0
C(194) = 0
C(195) = 0
C(196) = 0
C(197) = 0
C(198) = 0
C(199) = 0
C(200) = 0
C(201) = 0
C(202) = 0
C(203) = 0
C(204) = 0
C(205) = 0
C(206) = 0
C(207) = 0
C(208) = 0
C(209) = 0
C(210) = 0
C(211) = 0
C(212) = 0
C(213) = 0
C(214) = 0
C(215) = 0
C(216) = 0
C(217) = 0
C(218) = 0
C(219) = 0
C(220) = 0
C(221) = 0
C(222) = 0
C(223) = 0
C(224) = 0
C(225) = 0
C(226) = 0
C(227) = 0
C(228) = 0
C(229) = 0
C(230) = 0
C(231) = 0
C(232) = 0
C(233) = 0
C(234) = 0
C(235) = 0
C(236) = 0
C(237) = 0
C(238) = 0
C(239) = 0
C(240) = 0
C(241) = 0
C(242) = 0
C(243) = 0
C(244) = 0
C(245) = 0
C(246) = 0
C(247) = 0
C(248) = 0
C(249) = 0
C(250) = 0
C(251) = 0
C(252) = 0
C(253) = 0
C(254) = 0
C(255) = 0
C(256) = 0
C(257) = 0
C(258) = 0
C(259) = 0
C(260) = 0
C(261) = 0
C(262) = 0
C(263) = 0
C(264) = 0
C(265) = 0
C(266) = 0
C(267) = 0
C(268) = 0
C(269) = 0
C(270) = 0
C(271) = 0
C(272) = 0
C(273) = 0
C(274) = 0
C(275) = 0
C(276) = 0
C(277) = 0
C(278) = 0
C(279) = 0
C(280) = 0
C(281) = 0
C(282) = 0
C(283) = 0
C(284) = 0
C(285) = 0
C(286) = 0
C(287) = 0
C(288) = 0
C(289) = 0
C(290) = 0
C(291) = 0
C(292) = 0
C(293) = 0
C(294) = 0
C(295) = 0
C(296) = 0
C(297) = 0
C(298) = 0
C(299) = 0
C(300) = 0
C(301) = 0
C(302) = 0
C(303) = 0
C(304) = 0
C(305) = 0
C(306) = 0
C(307) = 0
C(308) = 0
C(309) = 0
C(310) = 0
C(311) = 0
C(312) = 0
C(313) = 0
C(314) = 0
C(315) = 0
C(316) = 0
C(317) = 0
C(318) = 0
C(319) = 0
C(320) = 0
C(321) = 0
C(322) = 0
C(323) = 0
C(324) = 0
C(325) = 0
C(326) = 0
C(327) = 0
C(328) = 0
C(329) = 0
C(330) = 0
C(331) = 0
C(332) = 0
C(333) = 0
C(334) = 0
C(335) = 0
C(336) = 0
C(337) = 0
C(338) = 0
C(339) = 0
C(340) = 0
C(341) = 0
C(342) = 0
C(343) = 0
C(344) = 0
C(345) = 0
C(346) = 0
C(347) = 0
C(348) = 0
C(349) = 0
C(350) = 0
C(351) = 0
C(352) = 0
C(353) = 0
C(354) = 0
C(355) = 0
C(356) = 0
C(357) = 0
C(358) = 0
C(359) = 0
C(360) = 0
C(361) = 0
C(362) = 0
C(363) = 0
C(364) = 0
C(365) = 0
C(366) = 0
C(367) = 0
C(368) = 0
C(369) = 0
C(370) = 0
C(371) = 0
C(372) = 0
C(373) = 0
C(374) = 0
C(375) = 0
C(376) = 0
C(377) = 0
C(378) = 0
C(379) = 0
C(380) = 0
C(381) = 0
C(382) = 0
C(383) = 0
C(384) = 0
C(385) = 0
C(386) = 0
C(387) = 0
C(388) = 0
C(389) = 0
C(390) = 0
C(391) = 0
C(392) = 0
C(393) = 0
C(394) = 0
C(395) = 0
C(396) = 0
C(397) = 0
C(398) = 0
C(399) = 0
C(400) = 0
C(401) = 0
C(402) = 0
C(403) = 0
C(404) = 0

```

77


```

1 IF (CNCE) 2,1,2
  CNCE = 1
  N1 = 1
  CPT = 0.0
  P1 = 2.251666666666667
  P2 = 2.258333333333333
  P3 = 2.254166666666667
  P4 = 2.25315
  C2 = 2.751666666666667
  C3 = 2.083333333333333
  C4 = 2.041666666666667
  KCUNT = 0
2 N1 = N1
  J2 = J1 + N1
  J3 = J2 + N1
  J4 = J3 + N1
  J5 = J4 + N1
  J6 = J5 + N1
  J7 = J6 + N1
  J8 = J7 + N1
  J9 = J8 + N1
  J10 = J9 + N1
  J11 = J10 + N1
  J12 = J11 + N1
  J13 = J12 + N1
  J14 = J13 + N1
  J15 = J14 + N1
  IF (CPT.NE.0.) GO TC 60
  IF (OPT) 60,49,60
  IF (DELT) 60,50,50
  IF (DELT) 50,200,50
  KCUNT = C
  GO 49
50
  START FUNGE-KUTTA INTEGRATION.
  COMPUTE KC
60 CONTINUE
  DO 25 I=1,N1
    K5 = J15 + I
    IF (SNGI(T(K5)) .NE. V(I+1)) T(K5) = V(I+1)
    IF (SNGI(T(K5)) - V(I+1)) 24,25,24
    T(K5) = V(I+1)
24 CONTINUE
25 IF (SNGI(TME) .NE. V(1)) TME = V(1)
  TME = V(1)
26 KCUNT = KCUNT + 1
27 DO 80 I=1,N1
  70 K0 = J7 + I

```

```

APRK0223C
APRK0224C
APRK0225C
APRK0226C
APRK0227C
APRK0228C
APRK0229C
APRK0230C
APRK0231C
APRK0232C
APRK0233C
APRK0234C
APRK0235C
APRK0236C
APRK0237C
APRK0238C
APRK0239C
APRK0240C
APRK0241C
APRK0242C
APRK0243C
APRK0244C
APRK0245C
APRK0246C
APRK0247C
APRK0248C
APRK0249C
APRK0250C
APRK0251C
APRK0252C
APRK0253C
APRK0254C
APRK0255C
APRK0256C
APRK0257C
APRK0258C
APRK0259C
APRK0260C
APRK0261C
APRK0262C
APRK0263C
APRK0264C
APRK0265C
APRK0266C
APRK0267C
APRK0268C
APRK0269C
APRK0270C
APRK0271C
APRK0272C
APRK0273C
APRK0274C
APRK0275C
APRK100C
APRK101C
APRK102C

```


AMRK163C
AMRK164C
AMRK165C
AMRK166C
AMRK167C
AMRK170C
AMRK171C
AMRK172C
AMRK173C
AMRK174C
AMRK175C
AMRK176C
AMRK177C
AMRK200C
AMRK201C
AMRK202C
AMRK203C
AMRK204C
AMRK205C
AMRK206C
AMRK207C
AMRK210C
AMRK211C
AMRK212C
AMRK213C
AMRK214C
AMRK215C
AMRK216C
AMRK217C
AMRK220C
AMRK221C
AMRK222C
AMRK223C
AMRK224C
AMRK225C
AMRK226C
AMRK227C
AMRK230C
AMRK231C
AMRK232C
AMRK233C
AMRK234C
AMRK235C
AMRK236C
AMRK237C
AMRK240C
AMRK241C
AMRK242C

```

DO 140 I=1,N1
K3=J10+I
140 T(K3)=C(I)*D(I+1)
C
C      COMPLETE VALUE OF FUNCTION.
C
DO 150 I=1,N1
K0=J7+I
K1=J8+I
K2=J9+I
K3=J10+I
K5=J15+I
T(K5)=T(K5)+0.16666666666666667*
1(T(KC)+T(K1)+T(K2)+T(K3))
V(I+1)=T(K5)
150 CONTINUE
170 CONTINUE
CALL ACXSUB
DO 180 I=1,N1
K5=J1+I
K0=J2+I
K1=J3+I
K2=J4+I
K3=J5+I
K4=J6+I
T(K4)=T(K3)
T(K3)=T(K2)
T(K2)=T(K1)
T(K1)=T(K0)
T(KC)=T(K5)
T(K5)=T(I)
T(I)=C(I+1)
180 CONTINUE
190 CONTINUE
RETURN
C
C      DO ACAMS-MCULTON INTEGRATION
C
200 CONTINUE
IF (KCUNT - 3) GO TO 60
IF (KCUNT = KCUNT + 1
201 KOUNT = C(I)*.5
DO 210 I=1,N1
K0=J7+I
K1=J2+I
K2=J3+I
K4=J1+I

```



```

C      K5=J15+I
C      CCMPLTE Y-PREDICTED.
C
210  T(K0)=T(K5)
      T(K5)=T(K0)+D(1)*(P1*T(I)-P2*T(K4)+P3*T(K1)-P4*T(K2))
      V(I+1)=T(K5)
      TME=TME+D(1)
      V(1)=TME
      CALL ALXSLB
      K5=0
      DO 220 I=1,N1
        K0=J8+I
        K2=J2+I
        K1=J1+I
        K4=J1+I
        K3=J15+I
C      CCMPLTE Y-CORRECTED
C
      T(K0)=T(K1)+D(1)*(P4*D(I+1)+C2*T(I)-C3*T(K4)+T(K2)*C4)
      ERF=14.*LMAX1(DABS(T(K1)),.01D0)
      ERF=ERF*EL(I)
      ERL=ERF*EL(I)
      TEMP=LABS(T(K0)-T(K2))
      IF (TEMP.LT.ERU) GO TO 215
      IF (TEMP-ERU) 215,214,214
      IF (ABS(SNGL(DELT))*.GF.LMIN) GO TO 260
      IF (ABS(SNGL(DELT))-HMIN) 215,260,260
      IF (AES(SNGL(DELT))-HMIN) 215,260,260
      CCNT=CCNT+1
      IF (TEMP.LT.ERL) K5=K5+1
      IF (TEMP-ERL) 216,22C,22C
      K5=K5+1
      CCNT=CCNT+1
      IF (K5.LT.N1) GO TO 300
      IF (K5-N1) 300,221,221
      IF (ABS(D(1)+D(1))*.GT.HMAX) GO TO 300
      IF (AES(D(1)+D(1))-HMAX) 222,222,300
C      SET-UP FOR DOUBLING STEP SIZE
C
      IF (KCUNT.LE.6) GO TO 300
      IF (KCUNT-6) 300,300,223
      CCNT=CCNT+1
      DO 240 I=1,N1
        K1=J1+I
        K2=J2+I
        K3=J3+I

```

```

AMRK243C
AMRK244C
AMRK245C
AMRK246C
AMRK247C
AMRK250C
AMRK251C
AMRK252C
AMRK253C
AMRK254C
AMRK255C
AMRK256C
AMRK257C
AMRK260C
AMRK261C
AMRK262C
AMRK263C
AMRK264C
AMRK265C
AMRK266C
AMRK272C
AMRK273C
AMRK274C
AMRK277C
AMRK300C
AMRK301C
AMRK302C
AMRK303C
AMRK304C
AMRK305C
AMRK306C
AMRK307C
AMRK310C
AMRK311C
AMRK312C
AMRK313C
AMRK314C
AMRK315C
AMRK316C

```


AMRK377C
AMRK4000
AMRK401C
AMRK4020
AMRK4030
AMRK404C
AMRK4050

```

36C V(I+1)=T(K1)
    TME=TME-C(I)*4.
    V(1)=TME
    D(1)=CELT
    CALL ALXSUB
    GO TC 50
    END

```

CC

```

SUBROUTINE PROCES
RETURN
END

```

CC

```

SUBROUTINE RESET
COMMON C(3415)
EQUIVALENCE (C(2300),NOLIST),(C(2301),LISTNC(1)),(C(2351),
1 VALUE(1))
EQUIVALENCE (C(2675),KRUN)
DIMENSION LISTING(50), VALUE(50)
IF(NOLIST.EQ.0) RETURN
DO 1 I=1,NCLIST
J=LISTNC(I)
1 C(J)=VALUE(I)
KRUN=KRUN+1
RETURN
END

```

C

CC C

INPT001C

```

SUBROUTINE INPT1
COMMON C(3415)
COMMON/CRUS/ ENG(30),IENG(15)
NAMELIST /MCNT/ ENG,IENG
EQUIVALENCE (C(2675), KRUN)
FLG1=0.C
IF (KRUN.GT.0) GO TC 100
READ (5,MCNT)
WRITE (6,MCNT)
10C RETURN
END

```

F 96

F 106
F 108

CC

```

SUBROUTINE INPT2

```

INPT001C

CLPT002C
CLPT001C
CLPT017C
CLPT003C
CLPT020C
CLPT0070
CLPT010C
CLPT011C
CLPT0120
CLPT013C
CLPT014C
CLPT0150
CLPT016C

CLPT0021C
CLPT0022C
CLPT0023C
CLPT0024C
CLPT0025C
CLPT0026C
CLPT0027C
CLPT0034C
CLPT0035C
CLPT0036C
CLPT0040C
CLPT0041C

```

SUBROUTINE CLPT3
  CUTPLT SUBFCUTINE CUFT3
  DIMENSION B(50),OUTNO(50),ONAME1(50),ONAME2(50)
  DIMENSION CNAME3(50)
  COMMON C(3215)
  INTEGER ITCNT,PGCNT,CUTNG
  EQUIVALENCE (C(2500),CUTNO(1))
  EQUIVALENCE (C(2550),GNAME1(1))
  EQUIVALENCE (C(2600),CNAME2(1))
  EQUIVALENCE (C(2150),GNAME3(1))
  EQUIVALENCE (C(2266C),DTCNT)
  EQUIVALENCE (C(22661),NCOU)
  EQUIVALENCE (C(22662),PGCNT)
  EQUIVALENCE (C(2667),ITCNT),(C(2668),PCNT),(C(2669),CPP)
  EQUIVALENCE (C(932),T)
  EQUIVALENCE (C(2913),DER)
  EQUIVALENCE (C(2670),TAPE),(C(2671),TAPEND)
  EQUIVALENCE (C(0290),XTP),(C(0298),YTP),(C(0306),ZTP)
  EQUIVALENCE (C(3400),PUNCH)
  EQUIVALENCE (C(0521),TSET)
  EQUIVALENCE (C(0522),PSI)
  EQUIVALENCE (C(0523),PHI)
  EQUIVALENCE (C(0554),AXECI)
  EQUIVALENCE (C(0555),AYECI)
  EQUIVALENCE (C(0556),AZECI)
  EQUIVALENCE (C(0577),VXECI)
  EQUIVALENCE (C(0552),VYECI)
  EQUIVALENCE (C(0553),VZECI)
  EQUIVALENCE (C(0547),XECI)
  EQUIVALENCE (C(0548),YECI)
  EQUIVALENCE (C(0549),ZECI)
  IF (ITCNT.GT.6) GC TO 7
  IF (ITCNT-6) 10,10,7
  IF ITCNT=ITCNT+1
  1C ITCNT=6.6) (I,C(I),C(I+1),C(I+2),C(I+3),C(I+4),C(I+5),C(I+6),
  1 C(I+7),I=1,3415,8)
  6 FORMAT(1H1/(15,2X,8E14.7))
  PGCNT=1
  7 PCNT=1
  8 IF (I.LE.PCNT) RETURN
  8 IF (I-PCNT) 12,9,9
  12 RETURN
  9 PCNT=PCNT+CPP
  9 IF (PGCNT.NE.1) GC TO 3
  9 IF (PGCNT-1) 3,1,2
  1 WRITE(6,2) (ONAME1(I),ONAME2(I),ONAME3(I), I = 1, NCOU)

```

C
C
C

C

C

C


```

PRGMM SETS LCONV = 2. STAGING WILL OCCUR WHEN THE VEHICLE IS
WITHIN + CR - ONE FCCT OF THE EARTH SURFACE, WHEN IT IS WITHIN + CR
- C.CC1 SECCND OF TF, OR ACCORDING TO SOME CRITERION IN ANOTHER
MODULE CR SUBROUTINE. FOR EXAMPLE, IF STAGING IS DESIRED AT
COINCIDENCE BETWEEN A AND B, THE FOLLOWING FORTRAN STATEMENTS
SHOULD BE USED IN ONE OF THE SUBROUTINES OF MODULES --

IF (ABS(A-B) .LE. 0.5) LCCNV = 2
IF ( (B-A) .LT. 0.) LCONV = 1
DIMENS C(100), RMS(10)
COMMON C(3415)
INTEGER STATNO
EQUIVALENCE (C(932), T), (C(2913), DER)
EQUIVALENCE (C(2668), PCNT)
EQUIVALENCE (C(5550), TF), (C(0507), H)
EQUIVALENCE (C(2904), KSTEP), (C(2905), STEP)
EQUIVALENCE (C(2670), TAPE), (C(2671), TAPEND)
EQUIVALENCE (C(2441), NCSTAT), (C(2200), STATNC(1)), (C(2430), RMS(1))
EQUIVALENCE (C(2442), LCSTAT)
EQUIVALENCE (C(2910), KCCNV)
EQUIVALENCE (C(2672), LCONV)
IF (ABS(H) .LE. 1.0) GO TO 2
IF ( (A) - 1.0) GO TO 1
IF ( (A) .LT. 0.6) GO TO 1
IF ( (A) .LT. 6) .LE. 0.0001) GO TO 2
IF (ABS(T-TF) - 0.0001) 2,2,7
IF (ABS(T-T) .LT. 0.) GO TO 1
IF ( (T-T) 1,8)
IF ( (T-T) 1,8) GO TO 2
IF (LCCNV - 2) 9,2,9
IF (LCCNV - EQ. 1) GO TO 1
IF (LCCNV - 1) 10,1,10
IF (LCCNV .LT. 0.) DER = -DER*0.5
IF (LCCNV - 11,12,12
IF (LCCNV - DER*0.5
IF (LCCNV - GT. 0.) DER = -DER*0.5
IF (LCCNV - 14,14,13
IF (LCCNV - DER*0.5
IF (LCCNV = KCCNV + 1
IF (KCCNV - GE. 10) GO TO 2
IF (KCCNV - 10) 15,2,2
IF (KCCNV - 1.0)
PCNT = 1.0
TAPEND = 1.0
CALL TAPE
K =

```

```

STGE0030
STGE0040
STGE0050
STGE0060
STGE0070
STGE0100
STGE0110
STGE0120
STGE0130
STGE0170
STGE0150
STGE0300
STGE0160
STGE0170
STGE0200
STGE0210
STGE0220
STGE0240
STGE0250
STGE0260
STGE0310
STGE0320
STGE0330
STGE0340
STGE0350
STGE0360
STGE0370
STGE0400
STGE0410
STGE0420
STGE0430
STGE0440
STGE0450
STGE0460
STGE0470
STGE0500
STGE0510
STGE0520
STGE0530
STGE0540
STGE0550
STGE0560
STGE0570
STGE0600
STGE0610
STGE0620
STGE0630

```


۷۷

۷۷

۷۷

66

UU

www. u


```

3 IF (NCRNDM) 2,2,3
  RNFLG=C
  DELT=DER
  DO 1 I=1,NCRNDM
    J=RNLMNC(I)
    C(J+5)=2.718281828*(-DER*C(J+4))
    C(J+6)=C(J+3)*SQRT(1.0-C(J+5)*C(J+5))
    ZN=C(J+1)
    Y=C(J)
    N=INT(Y)
    CALL RFG(N,X)
    C(J+7)=C(J+3)*X
    C(J+1)=ZN
  1 RETURN
  2 END

```

C C C

SUBROUTINE RNDM3

```

COMMON C(3415)
EQUIVALENCE (C(2443), RNFLG)
EQUIVALENCE (C(2444), RN)
EQUIVALENCE (C(2446), ZN)
EQUIVALENCE (C(2448), NCRNDM)
EQUIVALENCE (C(2449), DELT)
EQUIVALENCE (C(2450), RNDMNG)
EQUIVALENCE (C(2913), DER)
DIMENSION RNDMND(50)
INTEGER RNDMND
IF (NCRNDM) 20,20,1C
IF (RNFLG-GE.1.0) GC TC 8
IF (DER-GE.0.) GO TC 6
  8 DO 9 I=1,NCRNDM
    J=RNLMNC(I)
    X=(C(J+7)-C(J+5)*C(J+8))/C(J+6)
    IF (RNFLG-GE.1.0) GC TC 5
    RNFLG=1-C
    DELT=-DELT
    GO TC 1
  5 DELT=DELT+DER
  7 C(J+5)=2.718281828*(-DELT*C(J+4))
  5 C(J+6)=C(J+3)*SQRT(1.-C(J+5)*C(J+5))
  5 C(J+7)=C(J+6)*X+C(J+8)
  5 RETURN
  6 IF (DELT-EG.DER) GC TC 4
  6 DO 3 I=1,NCRNDM
    J=RNLMNC(I)

```



```

3 C(J+5)=2.718281828**(-DER*C(J+4))
  C(J+6)=C(J+3)*SQRT(1.0-C(J+5)*C(J+5))
4 DELT=DER
  DO 1 I=1,NCRNDM
    J=RANMNC(I)
    Y=C(J)
    ZN=C(J+1)
    N=INT(Y)
    CALL RFG(N,X)
    C(J+8)=C(J+7)
    C(J+7)=C(J+6)*X+C(J+5)*C(J+7)
    C(J+1)=ZN
1  C(J+1)=ZN
20 RETURN
   END

```

CC

```

SUBROUTINE AUXA1
RETURN
END

```

CC

```

SUBROUTINE AUXA2
RETURN
END

```

CC

```

SUBROUTINE AUXA3
RETURN
END

```

CC

```

SUBROUTINE AUXB1
RETURN
END

```

CC

```

SUBROUTINE AUXB2
RETURN
END

```

CC

```

SUBROUTINE AUXB3
RETURN
END

```

CC

```

SUBROUTINE AUXC1
RETURN

```

ALXA0010
ALXA0020
ALXA0030

ALXB0010
ALXB0020
ALXB0030

ALXB0010
ALXB0020
ALXB0030

ALXB0010
ALXB0020
ALXB0030

ALXC0010
ALXC0020

ALXC003C
ALXC001C
ALXC002C
ALXC003C
ALXC0010
ALXC002C
ALXC003C

```

END
SUBROUTINE ALXC2
RETURN
END
SUBROUTINE AUXC3
RETURN
END
SUBROUTINE RFG(M,X)
COMMON C(2415)
EQUIVALENCE (C(2446), ZN)
Y=FLCAT(N)
X=0.0
GENERATE UNIFORM RANDOM NO.(0 TO 1)
SUMFCR RAND NG(SIGMA=1,MEAN=0)
UNIFORM DIST FOR N=1
NORMAL DIST FOR LARGE N (12 OR GREATER)
DO 1 I=1,N
CO 1 I=1,N
X=X+ZN-0.5
RNC=557.*ZN
ZN=RND-AINT(RNC)
1 X=SQRT(12./Y)*X
RETURN
END

```

```

SUBROUTINE AII
AERODYNAMIC FORCES AND MOMENTS INITIALIZATION MODULE AII
BODY AXES
COMMON C(2415)
EQUIVALENCE (C(0208), TSA )
EQUIVALENCE (C(0257), CTS )
EQUIVALENCE (C(0258), STS )
CTS = CTS(TSA)
STS = SIN(TSA)
RETURN
END

```


END

SUBROUTINE A2

CAB - ZERC-LIFT BODY DRAG
CAC - DRAG CUE TO DEFLECTED CONTROL SURFACES
CAE - CAE+CAC
CZB - NORMAL FORCE DUE TO ANGLE OF ATTACK
CZC - NORMAL FORCE DUE TO FIN DEFLECTION
CYB - SIDE FORCE DUE TO SIDESLIP
CYC - SIDE FORCE DUE TO FIN DEFLECTION
CYD - SIDE FORCE
CMG - PITCH DAMPING MOMENT
CML - YAW DAMPING MOMENT
CLP - ROLL DAMPING MOMENT
CMB - PITCH MOMENT DUE TO ANGLE OF ATTACK
CMT - CCNTRCL SURFACE PITCH MOMENT
CME - CME+CMC
CMB - YAW MOMENT DUE TO SIDESLIP
CAC - CCNTRCL SURFACE YAW MOMENT
CNE - CNE+CNC
CLB - INDUCED ROLL MOMENT
CLC - CCNTRCL SURFACE ROLL MOMENT
CLL - CLB+CLC

COMMON/APPLE/TM1(7), TM2(15), TM3(14), TM4(5), TETA1(7), TETA2(7),
1 TETA3(6), TX11(2), TX12(10), TX13(5), TDEL(4), TH1(3), TH2(2),
2 TCZBC(7,2,7), TDCZP(14,7), TCYBC(7,10,6), TCMBC(7,5,7), TDCMF(14,7),
3 TCDB(15,2), TCDC(15,2), TCAD2(15), TCNBC(7,10,6), TCMO1(15,4),
4 TCMD2(5,4), TCMD(15), TCCMA(15), TPMD(14,3), TPQ(14,3),
5 TCLBC(7,10,6), TOLD(15), TPLD(14,3), TCLP(15), TPP(14,3), TDCYSP(3),
L TDCZSP(3), TDCMSP(3), TCNSP(3), TH3(3), TFEJ(13), TFEJ(13)
COMMON/ C(3415)
EQUIVALENCE (C(0016), DP)
EQUIVALENCE (C(0020), DG)
EQUIVALENCE (C(0024), DR)
EQUIVALENCE (C(0111), CA)
EQUIVALENCE (C(0113), CY)
EQUIVALENCE (C(0114), XI)
EQUIVALENCE (C(0115), CZ)
EQUIVALENCE (C(0116), CBRF)
EQUIVALENCE (C(0117), FTA)
EQUIVALENCE (C(0118), CMG)
EQUIVALENCE (C(0119), CAR)
EQUIVALENCE (C(0120), CLP)


```

VN=SCRT(VBA**2+WBA**2)
IF (VN.LT.C.00001) GT TC 10
XIR=ATAN2(WBA,WBA)
GO TC 20
1C XIR=C.C
20 XI=XIR*RAD
XI1=XI
IF (XI.LT.C.C) XI1=-XI
XI2=XI1
IF (XI1.GE.90.) XI2=180.-XI1
C TABLE LOCK-UP,ALL PHASES
CCZP=0.C
CCMP=C.C
PMC=1.C
FNC=1.C
PLC=1.C
PQ=1.C
PR=1.C
PP=1.C
CCNSEP=C.
DCNSEP=C.
DCLSEP=C.
CCYSEP=C.
DCZSEP=C.
DCLSP=C.
CCNSP=C.
CCYSP=C.
CCMENT C.CEFFICIENTS
FITTING MMENT
CMBC=TFREDL(AM,XI2,ETA,TM1,TXI3,TETAL,TCMBC,7,5,7)
CCMUL=CLLI(AM,TM2,TCMD,15)
CCMUL=CCNLL
CCNLL=-CCMUL
CCMLR=-CCMUL
CCMULA=CLLI(AM,TM2,TCNDA,15)
CCMURA=-CCMULA
CCMLLA=-CCMULA
CCMLRA=CCMULA
CMG=CLLI(AM,TM2,TCMC,15)
YAWING MMENT
CNBC=TFREDL(AM,XI2,ETA,TM1,TXI2,TETA3,TCNBC,7,10,6)
IF (XI1.GT.90.) CNBC=-CNBC
IF (XI.LT.C.C) CNBC=-CNBC
CCNUL=-CCMUL
CCNULF=CCMUL
CCNLL=-CCMUL
CCNLR=CCMUL
CCNULB=CCMULA

```



```

C
DCNURE=-DCMULA
DCNLLB=-DCMULA
DCNLRB=DCMULA
CNR=CMC
RCLLING NCMENT
CLBC=TPRECL(AM,XI2,ETA,TM1,TXI2,TETA3,TCLBC,7,10,6)
IF (XI1.GT.90.) CLBC=-CLBC
IF (XI1.LT.0.) CLBC=-CLBC
CLD=CDLI(AM,TM2,TCLD,15)
CLF=CELLI(AM,TM2,TCLF,15)
FCRCE CCEFFICIENTS
CYBCC=TPRECL(AM,XI2,ETA,TM1,TXI1,TETA1,TCZBC,7,2,7)
CYBCC=TPRECL(AM,XI2,ETA,TM1,TXI2,TETA3,TCYBC,7,10,6)
IF (XI1.GT.90.) CYBCC=-CYBCC
IF (XI1.LT.0.) CYBCC=-CYBCC
CAC2=CELLI(AM,TM2,TCAC2,15)
IF (XI1.GT.90.) GO TO 100
IF (XI1.LT.0.) GO TO 100
TABLE LOCK-UP, SEP AND BCOST PHASE
ZERC-LIFT CRAG
CDO=STCLIA(TM2,TH2,TCDB,AM,H,15,2)
IF (T.LT.1) CDO=CCC+.08
CCTRCL SURFACE EFFECTIVENESS
CMULC=STCLIA(TM2,TDELT,TCMD1,AM,ADUL,15,4)
IF (DUL.LT.0.0) CMULC=-CMULC
CMURC=STCLIA(TM2,TDELT,TCMD1,AM,ADUR,15,4)
IF (CUR.GT.0.0) CMURC=-CMURC
CMLLC=STCLIA(TM2,TDELT,TCMD1,AM,ADLL,15,4)
IF (CLL.LT.0.0) CMLLC=-CMLLC
CMRLC=STCLIA(TM2,TDELT,TCMD1,AM,ADLR,15,4)
IF (CLR.GT.0.0) CMRLC=-CMRLC
FLUNE EFFECTS
IF (AM.GE.1.6) GO TO 200
FOK=F/ICCC
FM=2.1212E-06*HOK**3-5.2017E-04*HOK**2+.044153*HOK-9.0909E-04
DCMP=STCLIA(TM3,TETA2,TCMP,AM,ETA,14,7)
IF (DCMP.LT.0.0) DCMP=0.0
DCZP=STCLIA(TM3,TETA2,TCZP,AM,ETA,14,7)
IF (DCZP.LT.0.0) DCZP=0.0
DCMP=DCMP*FM
DCZF=DCZF*FM
PMD=STCLIA(TM3,TH1,TFMD,AM,H,14,3)
IF (PMD.LT.0) PMD=1.0
PND=FM
PLD=STCLIA(TM3,TH1,TFLD,AM,H,14,3)
IF (PLD.LT.0) PLD=1.0
PQ=STCLIA(TM3,TH1,TFQ,AM,H,14,3)
IF (PQ.LT.0) PQ=1.0
PR=PC

```



```

CXI=CCS(XIR)
SXI=SSIN(XIR)
CAE=CAEC
CYB=CYEC
CZE=CYEC
CLE=CLBC
CNB=CNBC
CNTRCL SURFACE MOMENT IS 15.25 FT FROM NOSE
CONTRCL SURFACE CP IS 15.25-CGR)
F=(15.25-CG)/(15.25-CGR)
CMUL=CMULC+DCMULA*ALF*EUL+DCMUL*ADUL
CMLR=CMULC+DCMURA*ALF*CUR+DCMUR*ADUR
CMLL=CMULC+DCMLLA*ALP*CLL+DCMLL*ADLL
CMLR=CMULC+DCMLRA*ALP*CLR+DCMLR*ADLR
CMC=(CMUL+CMUR+CMLL+CMLR)*F*PMD
CNUL=CNULC+CNULB*BET*CUR+DCNUL*ADUL
CNUR=CNULC+CNURB*BET*CUR+DCNUR*ADUR
CNLL=CNULC+CNLLB*BET*DLL+DCNLL*ADLL
CNLR=CNULC+CNLRB*BET*CLR+DCNLR*ADLR
CNC=(CNUL+CNUR+CNLL+CNLR)*F*PND
CLUL=CLC*DUL
CLUR=CLE*EUR
CLLL=CLD*DLL
CLLR=CLE*DLR
CLC=(CLLL+CLUR+CLL+CLLR)*PLD
CZC=-CMC*CBAR/(15.25-CG)
CYC=-CNC*CEAR/(15.25-CG)
CAC=CAE2*(EUL**2+CUR**2+CLL**2+CLR**2+DLR**2+DUR**2+CLL**2+DLR**2+4.*1.3*(ALP*DELQ-BET*DELR))
TOTAL AERCDYNAMIC FORCE CCEFFICIENTS
CA=CAE+CAC
CY=CYB+CYC+CCYSEP
CZ=CZB+CZC+CCZSEP
CA=-CA
CZ=-CZ
TOTAL AERCDYNAMIC MOMENT CCEFFICIENTS
CL=CLE+CLC+CLLSEP
CLF=CLF*RAE*PP
CM=CNB+CMC+CZB*(CG-CGR)/CBAR+CCMSEP
CMG=CMG*RAE*PPQ
CN=CNB+CMC+CYB*(CG-CGR)/CBAR+CCNSEP
CNR=CNR*RAE*PR
RETURN
END

```



```

SUBROUTINE A3I
COMMON C(3415)
COMMON /FLAG/ FLG1
EQUIVALENCE (C(0601), FF1)
EQUIVALENCE (C(0602), WEIG)
EQUIVALENCE (C(0603), IA)
EQUIVALENCE (C(0604), FLG2)
FF1=C.C
WEIG=C.C
IA=0
FLG1=C.C
FLG2=C.C
RETURN
END

```

CC C CC C CC C CC C

SUBROUTINE A3

THIS IS THE MISSILE PROPUSSION MODULE

```

T1=BOCST ENGINE IGNITION TIME
BT=BOCST ENGINE BURNCUT TIME(=T3)
BE=CRUISE ENGINE IGNITION TIME(=T4)

```

```

COMMON /FLAG/ FLG1
COMMON C(3415)
EQUIVALENCE (C(0086), S)
EQUIVALENCE (C(0110), T)
EQUIVALENCE (C(0932), T1)
EQUIVALENCE (C(0933), T1)
EQUIVALENCE (C(0935), BT)
EQUIVALENCE (C(0936), BE)
EQUIVALENCE (C(0136), CG)
EQUIVALENCE (C(0150), OLT)
EQUIVALENCE (C(0151), CMT)
EQUIVALENCE (C(0152), CNT)
EQUIVALENCE (C(0153), EPSTH)
EQUIVALENCE (C(0154), PHITH)
EQUIVALENCE (C(0201), A)
EQUIVALENCE (C(0202), B)
EQUIVALENCE (C(0203), CC)
EQUIVALENCE (C(0073), TXBA)
EQUIVALENCE (C(0074), TYEA)
EQUIVALENCE (C(0075), TZE)
EQUIVALENCE (C(0507), F)
EQUIVALENCE (C(0520), AMACH)
EQUIVALENCE (C(0117), ETA)

```



```

EQUIVALENCE (C(0601), FF1)
EQUIVALENCE (C(0602), WFIG)
EQUIVALENCE (C(0603), IA)
EQUIVALENCE (C(0604), FLG2)
EQUIVALENCE (C(0605), DIFFM)
EQUIVALENCE (C(0606), FLGRJ)
EQUIVALENCE (C(0508), GD)
EQUIVALENCE (C(2913), CER)
DATA WFLCL/189./

C 80 FORMAT(1H,10X,'TOTAL ANGLE OF ATTACK EXCEEDED, ETA=',F8.3)
C
IF (FLG1.GT.0.0) GO TO 1C
CT = 0.0
DIFFM = C.
FF = 0.0
TXBA = C.0
TYBA = C.0
TZBA = 0.0
IF (T.LT.1) RETURN
FIN = F*12.*.0254
TIN = T
AIN = ETA
IF (T.GE.BE) GO TO 1
C BCCST PHASE
TT = T
IF (T.GT.BT) GO TO 10
CALL BCCST (TT, THRUS, FF)
TXBA = THRUS*CCS(EPSTH)
TNEA = THRUS*SIN(EPSTH)
TYBA = -TNEA*SIN(PHITH)
TZBA = TNEA*CCS(PHITH)
IB = 0
WEIGHT = WEIGHT-FF+FF1
CG = CG-(FF-FF1)/405.54
A = A-(FF-FF1)/256.25
B = B-(FF-FF1)/1.881
CC = B
CLT = C
CMT = TNEA*(15.525-CG)
CNT = -TYEA*(15.525-CG)
FF1 = FF
TW = T+DER
IF (TW.LT.BT) GO TO 2
IF (FLG2.GT.0.0) GO TO C 2
WEIGHT = WEIGHT-31.
CG = CG-.15
FLG2 = 1.0

```



```

GO TC 2
10 CONTINUE
IF (FLGRJ.GT.0.0) GO TO 21
CRUISE PHASE,RAMJET CFF
IF (IB.GE.1) GO TO 11
CALL ENGINE(HIN,AMACH,AIN,0.,TIN,1.0,0.0,0.,CT,SMARG,-1,1)
IB=1
11 IF (AIN.GT.9.3) AIN=9.3
CALL ENGINE(HIN,AMACH,AIN,0.,TIN,1.0,0.0,0.,CT,SMARG,1,1)
GO TC 4
C SIMPLIFIED RAMJET MODEL
C CRUISE PHASE,RAMJET CFF
21 CALL RAMJET (HIN,AMACH,AIN,FF,1.0,CT)
GO TC 4
1 CONTINUE
IF (FLGRJ.GT.0.0) GO TC 22
CRUISE PHASE,RAMJET CN
IB=0
IF (AIN.GT.10.0) GO TC 5
FF=FF1
IF (IA.GE.1) GO TC 3
CALL ENGINE (HIN,AMACH,AIN,FF,TIN,0.0,0.0,0.0,CT,SMARG,-1,1)
TT=1
FF1=FF
IA=1
XIN=0.0
IF (WEIG.GT.WFUEL) XIN=1.0
CALL ENGINE (HIN,AMACH,AIN,FF,TIN,XIN,0.0,0.0,CT,SMARG,1,1)
GO TC 4
C SIMPLIFIED RAMJET MODEL
C CRUISE PHASE,RAMJET CN
22 IF (AIN.GT.10.0) GO TC 5
XIN=C.C
IF (WEIG.GT.WFUEL) XIN=1.0
CALL RAMJET (HIN,AMACH,AIN,FF,XIN,CT)
GO TC 4
5 IF (FLG1.GT.0.0) GO TC 10
WRITE(C,EC) ETA
GO TC 10
TXBA=CT*GD*S
TYEA=0.
TZBA=C.
CT=T-TT
IF (FLGRJ.GT.0.0) GO TC 6
FF=FF/.45359237
C DIFFUSER (INLET) MARGIN IS NOT COMPUTED IN SIMPLIFIED RAMJET MODEL
C DIFFM=SMARG/100.

```



```

6  WEIG=WEIG+(ABS(FF+FF1))*CT/2.0
   IF (WEIG.GT.WFUEL) FLG1=1.0
   WEIGHT = WEIGHT - ABS(FF + FF1)*DT/2.
   TT=T
   FF1 = FF
   CG=CG-.C0026455*ABS(FF+FF1)*DT/2.
   A=A-.C031746*ABS(FF+FF1)*DT/2.
   B=B-.0063492*ABS(FF+FF1)*CT/2.
   CC=B
   RETURN
   ENC

```

2

C C C C C

```

SUBROUTINE A4I
ACTUATOR INITIALIZATION MODULE A4I

```

```

COMMON C(3415)
DIMENSION IPL(100)
EQUIVALENCE (C(1735),WFIN)
EQUIVALENCE (C(3315),N)
EQUIVALENCE (C(3316),IPL(1))
IF (WFIN.GT.C.0) GO TO 100
GO TO 200
100 IPL(N)=1653
   IPL(N+1)=1657
   IPL(N+2)=1701
   IPL(N+3)=1705
   IPL(N+4)=1709
   IPL(N+5)=1713
   IPL(N+6)=1717
   IPL(N+7)=1721
   N=N+8
   RETURN
200 END

```

100

200

C C C C C

```

SUBROUTINE A4
FIN ACTUATORS MODULE A4

```

```

COMMON C(3415)
EQUIVALENCE (C(0016),CP)
EQUIVALENCE (C(0020),CG)
EQUIVALENCE (C(0024),DR)
EQUIVALENCE (C(1693),CULD)

```



```

EQUIVALENCES (C(1696),DUL)
EQUIVALENCES (C(1697),CURC)
EQUIVALENCES (C(1700),CLR)
EQUIVALENCES (C(1701),CLLC)
EQUIVALENCES (C(1704),CLL)
EQUIVALENCES (C(1705),DLRC)
EQUIVALENCES (C(1708),CLR)
EQUIVALENCES (C(1709),CLD)
EQUIVALENCES (C(1712),D1)
EQUIVALENCES (C(1713),D2)
EQUIVALENCES (C(1716),D2)
EQUIVALENCES (C(1717),D3)
EQUIVALENCES (C(1720),D3)
EQUIVALENCES (C(1721),D4)
EQUIVALENCES (C(1724),D4)
EQUIVALENCES (C(1735),WFIN)
EQUIVALENCES (C(1736),DFIN)
EQUIVALENCES (C(1737),DGC)
EQUIVALENCES (C(1738),DRC)
EQUIVALENCES (C(1739),CPC)
EQUIVALENCES (C(1740),DMAX)
EQUIVALENCES (C(1743),DULDEG)
EQUIVALENCES (C(1734),CULDDG)

C ACTUATOR INPUT MIXER
  DULC=CGC+DRC+DPC
  DLRC=-CGC+DRC+DPC
  CLRC=-CGC-DRC+DPC
  FIN ANGLE CCMAND LIMITS
  IF (ABS(DULC).GT.DMAX) DULC=SIGN(DMAX,DULC)
  IF (ABS(DLRC).GT.DMAX) DLRC=SIGN(DMAX,DLRC)
  IF (ABS(DLRC).GT.DMAX) CLRC=SIGN(DMAX,CLRC)
  IF (ABS(DLRC).GT.DMAX) CLRC=SIGN(DMAX,DLRC)

C ACTUATOR DYNAMICS
  IF (WFIN.GT.0.0) GC TC 100
  GO TC 200
  SECCNC CORDER ACTUATORS
  RATE AND DEFLECTION LIMITS
  100 UPPER-LEFT FIN RESPONSE
    DIC=WFIN*WFIN*(DULC-DL)-2.*DFIN*WFIN*D1
    IF (ABS(D1).GE.RTLM) GC TO 110
    GO TC 120
  110 PROD=DIC*D1
  120 IF (PROD1.GE.0.0) DIC=0.0
    CULD=D1
    UPPER-RT FIN RESPONSE
    C2C=WFIN*WFIN*(DURC-DUR)-2.*DFIN*WFIN*D2

```



```

IF (ABS(C2).GE.RTLM) GC TO 130
GO TC 140
130 PRCD2=C2D*C2
140 IF (PRCD2.GE.0.0) C2D=0.0
C LOWER=LEFT FIN RESPONSE
C3C=WFIN*WFIN*(DULC-DLL)-2.*DFIN*WFIN*D3
IF (ABS(C3).GE.RTLM) GC TC 150
GO TC 160
150 PRCD3=C3C*C3
160 IF (PRCD3.GE.0.0) C3C=0.0
C CLLD=DT FIN RESPONSE
D4D=WFIN*WFIN*(DLRC-CLR)-2.*DFIN*WFIN*D4
IF (ABS(C4).GE.RTLM) GC TO 170
GO TC 180
170 PRCD4=C4D*C4
180 IF (PRCD4.GE.0.0) C4D=0.0
C *** IDEAL ACTUATORS ***
200 CUL=DULC
CUR=CURC
DLL=DLLC
CLR=CLRC
C FIN ANGLE LIMITERS
IF (ABS(DLL).GT.DMAX) DUL=SIGN(DMAX,DUL)
IF (ABS(DLR).GT.DMAX) DLR=SIGN(DMAX,DLR)
IF (ABS(DLL).GT.DMAX) DLL=SIGN(DMAX,DLL)
IF (ABS(DLR).GT.DMAX) DLR=SIGN(DMAX,DLR)
C PITCH, YAW, ROLL CHANNEL INPUTS
220 DQ=0.25*(CUL-CUR-CLR+DLL)
DR=0.25*(CUL+CUR-CLR-DLL)
DP=0.25*(CUL+CUR+CLR+DLL)
CULCEG=CUL*57.29578
CURCEG=CUR*57.29578
CLRCEG=CLR*57.29578
DLLCEG=DLL*57.29578
RETURN
END
C
C SUBROUTINE A51
RETURN
END
C
C SUBROUTINE A5
RETURN
END

```



```

C(1729),Y1D)
C(1732),Y1)
C(1737),CCE)
C(1738),CRC)
C(1739),LPC)
C(1740),CMAX)
C(0165),AKNZ)
C(0166),AKG)
C(0167),AKNY)
C(0168),AKR)
C(0169),AKF)
EQUIVALENCE
EQUIVALENCE
EQUIVALENCE
EQUIVALENCE
EQUIVALENCE
EQUIVALENCE
EQUIVALENCE
EQUIVALENCE
EQUIVALENCE
EQUIVALENCE
AKQI1=1.55
AKRI1=1.55
AKPI1=1.0
AKPI2=1.0
AKRI2=1.55
QCS=-2.75
CCE=1.45/57.2957795
IF (DYNP.GT.22.5) GC TC 3
AKNY=.1
AKNZ=.1
IF (DYNP.GT.4.0) GO TC 4
AKF=.8
AKR=1.1
AKG=1.1
GO TC 5
1 AKNY=.183502-.01573052*CYNP+3.183974E-04*CYNP**2-2.761728E-06*
1 IF (CYNP.GT.45.0) GC TC 5
AKNZ=.4536143-.02735*CYNP+6.307143E-04*CYNP**2-5.0E-06*CYNP**3
4 IF (CYNP.GT.35.0) GC TC 6
AKF=1.087-.1294302*CYNP+.007295*DYNP**2-1.935556E-04*DYNP**3
1 +1.533333E-06*CYNP**4
1 AKR=1.375087-.0819641*DYNP+.00252119*DYNP**2-3.00521E-05*DYNP**3
AKQ=2.75E-06*DYNP**4
1 +5.27E-06*DYNP**4
GC TC 6
5 AKNZ=.55625-.001225*CYNP
6 AKF=.1752143-.00285476*CYNP+1.47619E-05*DYNP**2
6 AKR=1.556357-.0665024*CYNP+.00112333*DYNP**2-6.66666E-06*DYNP**3
8 IF (CYNP.GT.33.0) GC TC 1
8 IF (CYNP.GT.2.0) GC TC 1
SEPARATION AND BOOST ACTCPILOT
GC=CCS
IF (T.CE.T1) QC=QCE
SI=1.0

```



```

S2=0.0
AKGI=AKGI1
AKRI=AKRI1
AKPI=AKPI1
GO TC 16
C CRUISE ALT CFI LCT
1 CONTINUE
S1=0.0
S2=1.0
AKGI=AKGI2
AKRI=AKRI2
AKPI=AKPI2
16 CONTINUE
FITCH ANC YAW CHANNELS
Q1=AKQ*(CN-S1*QC)
R1=AKR*RV
ANZM=-AZM/G
ANYM=AYM/G
ANZ=ANZC-ANZM
ANY=ANYC-ANYM
POS DGC AND DRC COMMAND NEG TURNING MOMENTS
CC1=Q1-S2*AKNZ*ANZ
ZIC=CC1
IF (ABS(LGC).GE.DMAX) ZID=0.0
LGC=CC1+AKGI*Z1
IF (ABS(LGC).GF.DMAX) DCC=SIGN(DMAX,DGC)
DRC1=R1-S2*AKNY*ANY
YIC=LRC1
IF (ABS(LRC).GE.DMAX) YID=0.0
LRC=LRC1+AKRI*Y1
IF (ABS(LRC).GF.DMAX) DRC=SIGN(DMAX,DRC)
2 CONTINUE
RCLL CHANNEL
PLC=AKPI*FM
LAG COMPENSATOR
CMG1=.1
CMG2=.1
LPCG=-AKP*(FM+P1)*OMG2/CMG1
LPCIC=(CMG1-CMG2)*CF(C-OMG2*DPCI
DPC=DPCI+DPCG
IF (ABS(DPC).GF.DMAX) DPC=SIGN(DMAX,DPC)
RETURN
END
SUBROUTINE C2I

```



```

DIMENSION IPL(100)
COMMON C(3415),
EQUIVALENCE (C(0507),F)
EQUIVALENCE (C(3315),N)
EQUIVALENCE (C(3316),IPL(1))
IPL(N)=1633
IPL(N+1)=1637
N=N+2
HNCW=H
RETURN
END

```

C C C C C

```

SUBROUTINE C2
GLIDANCE COMMAND MODULE

```

```

COMMON C(3415)
EQUIVALENCE (C(0256), A33)
EQUIVALENCE (C(0286), VXTP)
EQUIVALENCE (C(0294), VYTP)
EQUIVALENCE (C(0302), VZTP)
EQUIVALENCE (C(0337), CMGYC)
EQUIVALENCE (C(0317), CMGZC)
EQUIVALENCE (C(0317), FLGT)
EQUIVALENCE (C(0339), XMA)
EQUIVALENCE (C(0340), YMA)
EQUIVALENCE (C(0341), ZMA)
EQUIVALENCE (C(0520), AMACH)
EQUIVALENCE (C(0500), FCC)
EQUIVALENCE (C(0501), FREF)
EQUIVALENCE (C(0502), HD)
EQUIVALENCE (C(0507), F)
EQUIVALENCE (C(0508), CC)
EQUIVALENCE (C(0515), PSF)
EQUIVALENCE (C(0527), GAMMAV)
EQUIVALENCE (C(0528), VT)
EQUIVALENCE (C(0531), VF)
EQUIVALENCE (C(0581), CYNP)
EQUIVALENCE (C(0582), AKG)
EQUIVALENCE (C(0932), T1)
EQUIVALENCE (C(0933), T2)
EQUIVALENCE (C(0935), T3)
EQUIVALENCE (C(0937), T5)
EQUIVALENCE (C(0942), TCC)
EQUIVALENCE (C(0944), TLD)
EQUIVALENCE (C(0945), TFC)

```



```

EQUIVALENCE (C(1633), ANZCD)
EQUIVALENCE (C(1636), ANZC)
EQUIVALENCE (C(1637), ANYCD)
EQUIVALENCE (C(1640), ANYC)
EQUIVALENCE (C(1750), WFI)
DATA RCCN/36500./
DYNP=CC/144.
C ZERC CCNTROL SWITCHES
S4=0.0
S5=C.C
S6=C.C
S8=0.0
IF (T.GE.T3) GO TC 1
C SEPARATION: BCCST PHASE TC 1
IF (AMACF.GE.2.0) GC TC 1
GO TC 100
C ZERC TRAJECTORY
1 IF (T.GE.T5) GO TO 2
GO TC 100
C LEVEL FLIGHT
2 IF (T.GE.TDC) GO TO 5
S4=1.0
S5=1.0
GO TC 100
C DIVE: CLIMB: TLC) GO TO 4
5 IF (T.GE.TLC) GO TO 4
7 HDCI=HDC
S4=1.0
S5=1.0
S8=1.0
GO TC 100
C AUTOMATIC LETDCWN TO HREF
4 FORI2=SCRT(XMA**2+YMA**2)
D1=1.975*VF+RCCN-HCRI2
IF (D1.GE.C.C) GO TC 6
S4=1.0
S5=1.0
S6=1.0
GO TC 100
C TERMINAL CIVE
6 W3=(ZMA*V)TP-XMA*VZTP)/(XMA*VXTP+ZMA*VZTP)
ANZC1=-2.*2900.*W3/32.172+A33
IF (FLGT.GT.0.0) GC TC 8
GO TC 100
C TERMINAL FCING
8 ANZC1=AKG*VT*CMGYC/32.174
GO TC 100
100 CCNTINLE

```



```

C ***** VERTICAL CHANNEL *****
C HERR LIMITER
  HERR1=SC*(F-FREF)
  HERR2=C.5*HERR1
  IF (ABS(HERR1).GT.8CC.) HERR2=SIGN(400.,HERR1)
  ANZC1=.0333*(S8*HDC1-S4*(HD+HERR2))+S5*A33
C NZ LIMITER
  IF(ANZC1) 108,107,1CE
  IF(AZMAX=.35*(CYNP-5.714)
  IF(DYNP.LE.30.0) AZMAX=8.5*DYNP/30.
  AZMAX=AMIN1(AZMAX,12.C)
  IF(ABS(ANZC1).GE.AZMAX) ANZC1=SIGN(AZMAX,ANZC1)
C 107 CONTINUE
C ***** LATERAL CHANNEL *****
  ANYC1=C.
  IF(T.LT.15) GO TO 21C
  IF(FLGT.GT.0.0) GC TC 200
C CROSS-PRCCUCT MIDCOURSE GUIDANCE
  CPROD=(YMA*VXTP-XMA*VYTP)/(XMA*VXTP+YMA*VYTP)
  ANYC1=2900.*CPROD/32.172
  GO TC 205
C TERMINAL PCMING
  200 ANYC1=AKG*VT*GGMGZC/32.172
C NY LIMITER
  205 IF (ANYC1) 208,207,208
  208 AYMAX=E.C*(DYNP-11.25)/22.5
  IF (CYNP.LE.22.5) AYMAX=4.0*DYNP/22.5
  AYMAX=AMIN1(AYMAX,12.0)
  IF (ABS(ANYC1).GT.AYMAX) ANYC1=SIGN(AYMAX,ANYC1)
C 207 CONTINUE
C ELIFITICAL LIMITING
  DENOM=SCRT((AZMAX*ANYC1)**2+(AYMAX*ANZC1)**2)
  IF (DENOM.LE.0.0) GC TO 21C
  AZLIM=AZMAX*X*AYMAX*ANZC1/CENOM
  IF (ABS(ANZC1).GE.ABS(AZLIM)) ANZC1=AZLIM
  AYLIM=AZMAX*AYMAX*ANYC1/CENOM
  IF (ABS(ANYC1).GE.ABS(AYLIM)) ANYC1=AYLIM
C LATERAL CCMAND FILTER
  210 ANYCC=WFI*(ANYC1-ANYC)
C VERTICAL CCMAND FILTER
  ANZCC=WFI*(ANZC1-ANZC)
  RETCFN
  END
C
C SUBROUTINE C2I
  RETURN
  END

```


CC	SUBROUTINE C3 RETURN END
CC	
CC	SUBROUTINE C4I RETURN END
CC	
CC	SUBROUTINE C4 RETURN END
CC	
CC	SUBROUTINE C5I RETURN END
CC	
CC	SUBROUTINE C5 RETURN END
CC	
CC	SUBROUTINE C6I RETURN END
CC	
CC	SUBROUTINE C6 RETURN END
CC	
CC	SUBROUTINE C7I RETURN END
CC	
CC	SUBROUTINE C7 RETURN END
CC	
CC	SUBROUTINE C8I

CC C CC C

```

SUBROUTINE D1
TRANSLATIONAL DYNAMICS MODULE D1B - BODY AXES

CCMMCN C(3415)
EQUICIVALENCE (C(0073) , TXBA )
EQUICIVALENCE (C(0074) , TYBA )
EQUICIVALENCE (C(0075) , TZBA )
EQUICIVALENCE (C(0086) , WE )
EQUICIVALENCE (C(0112) , FEJ )
EQUICIVALENCE (C(0128) , FXBA )
EQUICIVALENCE (C(0129) , FYBA )
EQUICIVALENCE (C(0130) , FZBA )
EQUICIVALENCE (C(0224) , A11 )
EQUICIVALENCE (C(0228) , A12 )
EQUICIVALENCE (C(0232) , A13 )
EQUICIVALENCE (C(0236) , A21 )
EQUICIVALENCE (C(0240) , A22 )
EQUICIVALENCE (C(0244) , A23 )
EQUICIVALENCE (C(0248) , A31 )
EQUICIVALENCE (C(0252) , A32 )
EQUICIVALENCE (C(0256) , A33 )
EQUICIVALENCE (C(0282) , AG )
EQUICIVALENCE (C(0283) , VXDTP )
EQUICIVALENCE (C(0286) , VXTIP )
EQUICIVALENCE (C(0287) , XDTP )
EQUICIVALENCE (C(0290) , XTP )
EQUICIVALENCE (C(0291) , YDTP )
EQUICIVALENCE (C(0294) , VYTP )
EQUICIVALENCE (C(0295) , YTP )
EQUICIVALENCE (C(0298) , YDTP )
EQUICIVALENCE (C(0299) , VZDTP )
EQUICIVALENCE (C(0302) , VZTP )
EQUICIVALENCE (C(0303) , ZDTP )
EQUICIVALENCE (C(0306) , ZTP )
EQUICIVALENCE (C(0411) , GCX )
EQUICIVALENCE (C(0412) , GCY )
EQUICIVALENCE (C(0413) , GCZ )
EQUICIVALENCE (C(0710) , TAXBA )
EQUICIVALENCE (C(0711) , TAYBA )
EQUICIVALENCE (C(0712) , TAZBA )
EQUICIVALENCE (C(0713) , TAX )
EQUICIVALENCE (C(0714) , TAY )
EQUICIVALENCE (C(0715) , TAZ )
AGWE = AG/WE

```



```

C      TOTAL ACCELERATION DUE TO AERODYNAMIC FORCES AND THRUST FORCES
TAXBA = AGWE*(FXBA + TXBA)
TAYBA = AGWE*(FYBA + TYBA)
TAZBA = AGWE*(FZBA + TZBA + FEJ)
RESOLVE FROM BODY AXES TO TANGENT PLANE
TAX = A11*TAXBA + A21*TAYBA + A31*TAZBA
TAY = A12*TAXBA + A22*TAYBA + A32*TAZBA
TAZ = A13*TAXBA + A23*TAYBA + A33*TAZBA
INTEGRATE ACCELERATIONS DUE TO AERODYNAMICS, THRUST, GRAVITY,
AND CORIOLIS
VXDTF = TAX + GCX
VYDTF = TAY + GCY
VZDTF = TAZ + GCZ
INTEGRATE VELOCITY
XDTP = VXTP
YDTP = VYTP
ZDTP = VZTP
RETURN
END

```

```

SUBROUTINE D2I
ROTATIONAL DYNAMICS INITIALIZATION MODULE D2IEUL
FOR USE WITH MODULE C2F CR D2SA

```

```

COMMON C(3415)
EQUIVALENCE (C(224), A11)
EQUIVALENCE (C(228), A12)
EQUIVALENCE (C(232), A13)
EQUIVALENCE (C(236), A21)
EQUIVALENCE (C(240), A22)
EQUIVALENCE (C(244), A23)
EQUIVALENCE (C(248), A31)
EQUIVALENCE (C(252), A32)
EQUIVALENCE (C(256), A33)
EQUIVALENCE (C(2901), ROLL)
EQUIVALENCE (C(2902), PITCH)
EQUIVALENCE (C(2903), YAW)
EQUIVALENCE (C(3315), N)
EQUENSICN IPL (100)
SP=SIN(ROLL)
CT=CCS(PITCH)
ST=SIN(PITCH)
EQU=VALENCE (C(3316), IPL(1))
SS=CCS(YAW)

```



```

CP = CS(RCLLO)
A11 = CS*CT
A12 = CS*CT
A13 = CS*ST*CP + CS*ST*SF
A21 = CS*CP + SS*ST*SP
A22 = CS*CP + SS*ST*SP
A31 = CS*ST*CP + SS*SF
A32 = CS*ST*CP - CS*SP
A33 = CS*CT*CP
IPL(N) = 2CS
IPL(N+1) = 213
IPL(N+2) = 217
IPL(N+3) = 221
IPL(N+4) = 225
IPL(N+5) = 229
IPL(N+6) = 233
IPL(N+7) = 237
IPL(N+8) = 241
IPL(N+9) = 245
IPL(N+10) = 249
IPL(N+11) = 253
N = N+12
N RETURN
END

```

CCC CCC

SUBROUTINE C2

ROTATIONAL DYNAMICS MODULE D2PA- PRINCIPAL AXES

```

COMMON C(2415)
COMMCN/APPLE/TM1(7), TM2(15), TM3(14), TM4(5), IETA1(7), IETA2(7),
1 IETA3(6), TXI1(2), TXI2(10), TXI3(5), TDEL(4), TFI(3), TH2(2),
2 TCZBC(7,2), TDCZP(14,7), TCYBC(7,10,6), TCMBC(7,5,7), TDCMF(14,7),
3 TCDB(15,2), TDC(15,2), TCAD2(15), TCNRC(7,10,6), TCMOI(15,4),
4 TCMD2(5,4), TCMD(15), TDCMDA(15), TPMD(14,3), TCMQ(15), TPQ(14,3),
5 TCLBC(7,10,6), TCLD(15), TPLD(14,3), TCLP(15), IFP(14,3), TDCYSP(3),
6 TDCZSP(3), TDCMSP(3), TH3(3), TFEJ(13), TFEJ(13)
EQUIVALENCE(ENCB, (C(0109), FLGSEP)
EQUIVALENCE(ENCB, (C(0112), FEJ)
EQUIVALENCE(ENCB, (C(0131), CLBA)
EQUIVALENCE(ENCB, (C(0132), CMBA)
EQUIVALENCE(ENCB, (C(0133), CNBA)
EQUIVALENCE(ENCB, (C(0136), CG)
EQUIVALENCE(ENCB, (C(0150), CLT)
EQUIVALENCE(ENCB, (C(0151), CMT)

```



```

AD22=A32*PBA-A12*RBA
AD21=A31*PBA-A11*RBA
AD13=A23*RBA-A33*QBA
AD12=A22*RBA-A32*QBA
AD33=A13*GBA-A23*PBA
AD32=A12*GBA-A22*PBA
AD31=A11*GBA-A21*PBA
AD23=A13*PBA-A13*RBA
RETURN
END

```

CC

```

SUBROUTINE D3I
RETURN
END

```

CC

```

SUBROUTINE D3
RETURN
END

```

CC

```

SUBROUTINE D4I
RETURN
END

```

CC

```

SUBROUTINE D4
RETURN
END

```

CC

```

SUBROUTINE D5I
RETURN
END

```

CC

```

SUBROUTINE D5
RETURN
END

```

CC

```

SUBROUTINE G1I

```

CC

GRAVITATIONAL AND CIRCULAR ACCELERATION INITIALIZATION MODULE G1IC

```

COMMON (13415)

```

CC


```

C      EQUIVALENCE (C(0507), F, )
C      EQUIVALENCE (C(0507), F, )
C      IF GZRC = C, FLAT-EARTH GRAVITATIONAL FIELD
C      IF (GZRC .EQ. 1.0) GC TO 1
C      COMPUTE FLAT-EARTH GRAVITATIONAL ACCELERATION
C      GCX = C.
C      GCY = C.
C      GCZ = GC + FC
C      F = -ZTP + FC
C      FD = -VZTP
C      RETURN
C      1 IF GZRC = 1, SPHERICAL GRAVITATIONAL FIELD
C      R = SQRT(XTP*XTTP + YTP*YTTP + (ZTP-ZS)*(ZTP-ZS))
C      FD = (XTF*VXTF + YTF*VYTF + (ZTP-ZS)*VZTP)/R
C      F = R - RE
C      COMPUTE SPHERICAL GRAVITATIONAL ACCELERATION
C      GMR = -GM/(R*R*R)
C      GX = GMR*XTTP
C      GY = GMR*YTTP
C      GZ = GMR*(ZTP-ZS)
C      ADD CCRLIS ACCELERATION CMGY*VZTP
C      GCX = GX + CMGZ*VYTF - CMGZ*VXTTP
C      GCY = GY + CMGX*VZTP - CMGX*VYTP
C      GCZ = GZ + CMGY*VXTTP - CMGX*VYTP
C      RETURN
C      END
C
C      SUBROUTINE G2I
C      RETURN
C      END
C
C      SUBROUTINE G2
C      RETURN
C      END
C
C      SUBROUTINE G3I
C      COMMON C(3415)
C      EQUIVALENCE (C(0208), TSA )
C      EQUIVALENCE (C(0257), CTS )
C      EQUIVALENCE (C(0258), STS )
C      EQUIVALENCE (C(0306), ZTP)
C      EQUIVALENCE (C(0414), FC)
C      EQUIVALENCE (C(0507), H)

```



```

EQUIVALENCE (C(0515), FSF)
EQUIVALENCE (C(0529), VAT)
EQUIVALENCE (C(0551), RICH)
EQUIVALENCE (C(0561), VAF)
F=FC-ZTF
ALT=F/3*280839895
VEL=VAT/3*280839895
CALL AIR (ALT,VEL,-1,P,T,SON,DEN,Q,G,1,2)
CTS = CCS(TSA)
STS = SIN(TSA)
RETURN
END

```

CC C CC C

```

SUBROUTINE G2
AIR DATA MODULE G3
CGMCMCN C(3415)
EQUIVALENCE (C(0224), A11)
EQUIVALENCE (C(0228), A12)
EQUIVALENCE (C(0232), A13)
EQUIVALENCE (C(0236), A21)
EQUIVALENCE (C(0240), A22)
EQUIVALENCE (C(0244), A23)
EQUIVALENCE (C(0248), A31)
EQUIVALENCE (C(0252), A32)
EQUIVALENCE (C(0256), A33)
EQUIVALENCE (C(0257), CTS)
EQUIVALENCE (C(0258), STS)
EQUIVALENCE (C(0282), AG)
EQUIVALENCE (C(0286), VXT)
EQUIVALENCE (C(0294), VYTP)
EQUIVALENCE (C(0302), VZTP)
EQUIVALENCE (C(0451), TWXTP)
EQUIVALENCE (C(0452), TWYTP)
EQUIVALENCE (C(0453), TWZTP)
EQUIVALENCE (C(0504), VATX)
EQUIVALENCE (C(0505), VATY)
EQUIVALENCE (C(0506), VATZ)
EQUIVALENCE (C(0507), F)
EQUIVALENCE (C(0508), GD)
EQUIVALENCE (C(0509), LBA)
EQUIVALENCE (C(0510), VBA)
EQUIVALENCE (C(0511), WBA)
EQUIVALENCE (C(0512), USA)
EQUIVALENCE (C(0513), VSA)

```



```

C VELCCITY IN BODY AXES
C UBA=A11*VATX+A12*VATY+A13*VATZ
C WBA=A21*VATX+A22*VATY+A23*VATZ
C WBA=A31*VATX+A32*VATY+A33*VATZ
C VELCCITY IN STABILITY AXES
C LSA=CTS*UBA+STS*WBA
C VSA=VEA
C WSA=STS*UBA+CTS*WBA
C ANGLE CF ATTACK AND SIDESLIP
C ASA = ATAN(WSA/USA)
C BSA = ARCSIN(VSA/SORT(LSA*USA + VSA*VSA + WSA*WSA))
C RETURN
C END

SUBROUTINE G4I
C RETURN
C END

SUBROUTINE G4
C RETURN
C END

SUBROUTINE G5I
C RETURN
C END

COORDINATE CCVERSION INITIALIZATION MODULE

COMMON C(3415)
C EQUIVALENCE (C(0557),CTPI11)
C EQUIVALENCE (C(0558),CTPI12)
C EQUIVALENCE (C(0559),CTPI13)
C EQUIVALENCE (C(0560),CTPI21)
C EQUIVALENCE (C(0562),CTPI22)
C EQUIVALENCE (C(0563),CTPI23)
C EQUIVALENCE (C(0564),CTPI31)
C EQUIVALENCE (C(0565),CTPI32)
C EQUIVALENCE (C(0566),CTPI33)
C EQUIVALENCE (C(0576),ALAT)
C EQUIVALENCE (C(0578),AZ)
C AZ = AZ*.C1745329

```



```

ELEV = ATAN (-ZTP/SCFT(XTF*YTP + YTP*YTP))
GO TC 6C
ELEV=0.C
50 CONTINUE
6C C61 = CTF1*CTP
C CBI11 = CTF111*CTPB11 + CTF112*CTPB12 + CTF113*CTPB13
C CBI12 = CTF111*CTPB21 + CTF112*CTPB22 + CTF113*CTPB23
C CBI13 = CTF111*CTPB31 + CTF112*CTPB32 + CTF113*CTPB33
C CBI21 = CTF121*CTPB11 + CTF122*CTPB12 + CTF123*CTPB13
C CBI22 = CTF121*CTPB21 + CTF122*CTPB22 + CTF123*CTPB23
C CBI23 = CTF121*CTPB31 + CTF122*CTPB32 + CTF123*CTPB33
C CBI31 = CTF131*CTPB11 + CTF132*CTPB12 + CTF133*CTPB13
C CBI32 = CTF131*CTPB21 + CTF132*CTPB22 + CTF133*CTPB23
C CBI33 = CTF131*CTPB31 + CTF132*CTPB32 + CTF133*CTPB33
C CETERMINE ACCELERATION, VELOCITY AND POSITION IN ECI SYSTEM
AXECI = CTF111*TAXAA + CTF112*TAYAA + CTF113*TAZAA
AYECI = CTF121*TAXAA + CTF122*TAYAA + CTF123*TAZAA
VXECI = CTF111*VXTTP + CTF112*VYTP + CTF113*VZTP
VYECI = CTF121*VXTTP + CTF122*VYTP + CTF123*VZTP
VZECI = CTF131*VXTTP + CTF132*VYTP + CTF133*VZTP
XECI = CTF111*XTP + CTF112*YTP + CTF113*(ZTP - ZS)
YECI = CTF121*XTP + CTF122*YTP + CTF123*(ZTP - ZS)
ZECI = CTF131*XTP + CTF132*YTP + CTF133*(ZTP - ZS)
RETURN
END

CC SUBROUTINE G6I
RETURN
END

CC SUBROUTINE G6
RETURN
END

CC SUBROUTINE S1I
RETURN
END

CC CC HCNING SEEKER INITIALIZATION MODULE S1I
COMMON C(3415)
DIMENSION IPL(100)
EQUIVALENCE (C(0317), FLGT)
EQUIVALENCE (C(0347), FLGTS)
EQUIVALENCE (C(3315), N)
EQUIVALENCE (C(3316), IPL(1))

```



```

IPL(N)=26266
IPL(N+1)=270
IPL(N+2)=274
IPL(N+3)=272
IPL(N+4)=226
IPL(N+5)=416
IPL(N+6)=420
IPL(N+7)=424
IPL(N+8)=428
IPL(N+9)=428
N=N+10
FLGT=C.C
FLGTS=C.O
RETURN
END

```

C C C C C C

```

SUBROUTINE S1
  HCMING SEEKER MODULE S1
  CCMMCN C(3415)
  EQUIVALENCE (C(187), CNGIS1)
  EQUIVALENCE (C(197), CNGIS2)
  EQUIVALENCE (C(212), P)
  EQUIVALENCE (C(216), Q)
  EQUIVALENCE (C(220), R)
  EQUIVALENCE (C(262), CMGYD)
  EQUIVALENCE (C(265), CMGY)
  EQUIVALENCE (C(266), CMGZD)
  EQUIVALENCE (C(269), CMGZ)
  EQUIVALENCE (C(270), THTS)
  EQUIVALENCE (C(273), THTS)
  EQUIVALENCE (C(274), PSISD)
  EQUIVALENCE (C(277), PSIS)
  EQUIVALENCE (C(279), CMGYS)
  EQUIVALENCE (C(280), ANGLT)
  EQUIVALENCE (C(316), RS)
  EQUIVALENCE (C(317), FLGT)
  EQUIVALENCE (C(318), RMT)
  EQUIVALENCE (C(319), XLT)
  EQUIVALENCE (C(320), YDT)
  EQUIVALENCE (C(321), ZDT)
  EQUIVALENCE (C(322), CMZCID)
  EQUIVALENCE (C(325), CMZCI)
  EQUIVALENCE (C(326), CMYCI)
  EQUIVALENCE (C(329), CMYCI)

```



```

C      AZA=ATAN(YCA/RXZA)
KEEP SEEKER (AGED IF POINTING ERRORS EXCEED GIMBAL LIMITS
ELAI=ELA
AZAI=AZA-4./RAD
IF (ABS(ELAI).GT.GMBLIM) GO TO 200
IF (ABS(AZAI).GT.GMBLIM) GO TO 200
PRECESSEEK TO SEARCH INITIATION PT
CMGYC=AKS*ELAI
CMGZC=AKS*AZA1
GO TO 50
C      SEARCH SEQUENCE
10 IF (FLAGS.GT.0.0) GC TC 20
TIME=T
FLAGS=1.C
WRITE (6,150) RMA,T
20 IF (FLGTS.GT.0.0) GC TO 75
TS = T-TIME
IF (TS.GT.0.4) GO TC 30
CMGYC=-10./RAD
CMGZC=C
CMGC=ABS(CMGYC)
GO TC 50
30 IF (TS.GT.1.35) GC TC 40
CMGYC=C
CMGZC=1C./RAD
CMGC=ABS(CMGZC)
GO TC 50
40 IF (TS.GT.1.775) GC TC 50
CMGYC=TRLIM
CMGZC=C
CMGC=ABS(CMGYC)
GO TC 50
50 IF (TS.GT.2.4) GO TC 60
CMGYC=C
CMGZC=-TRLIM
CMGC=ABS(CMGZC)
GO TC 50
60 IF (TS.GT.2.625) GO TC 65
CMGYC=-TRLIM
CMGZC=C
CMGC=ABS(CMGYC)
GO TC 50
65 IF (TS.GT.3.425) GO TC 70
CMGYC=C
CMGZC=TRLIM
CMGC=ABS(CMGZC)
GO TC 50
C      END CF SEARCH

```



```

70 IF (FLGL.GT.0.0) GO TC 90
   CMGZC=C.C
   CMGYC=C.C
   WRITE (6,130)
   FLGTS=1.C
   GO TC 90
75 IF (FLGL.GT.0.0) GC TC 80
   GO TC 90
80 REPCSTICN SEEKER ON DETECTED TARGET
80 IF (FLGT.GT.0.0) GC TC 85
   RYZT=SGRT(YCT**2+ZDT**2)
   CMGYC=-TRLIM*ZDT/RYZT
   CMGZC=TRLIM*YDT/RYZT
   IF (ANGLT.LE.0.25) FLGT=1.0
   GO TC 90
C   RADIOMETER AND SENSOR FILTER LAGS
85 WS=201.
   WSF=125.*(.2*ERSEL-D1Y)
   D1YD=WS* (.2*ERSAZ-D1Z)
   RECEIVER=NCISE MODEL
   SGMNOI=3.C4815E-05*RMT*2.718282**(RMT*5.614624E-06)
   FLNCIS=0.2*SGMNOI*CNCSIS1
   AZNOI=C.2*SGMNOI*CNCSIS2
   ADD NCISE TO FILTERED RADIOMETER SIGNAL
   D1Y=D1Y+ELNCIS
   D1ZN=D1Z+AZNOIS
   CMYCID=WSF*(AKS*D1YN/RAD-OMYC1)
   CMZCID=WSF*(AKS*D1ZN/RAD-OMZC1)
   CMGZC=OMZC1
   CMGYC=CMYC1
C   IF SEEKER LCSES TGT, PRECESSION COMMANDS GO TC ZERO
   IF (ABS(ANGLT).LE.2.5) GC TO 90
88 IF (FLAGLT.GT.0.0) GC TC 90
   WRITE (6,140) ANGLT
   FLAGLT=1.0
C   PRESSION RATES
   SEEKER FR (CMGYC).GE.TRLIM) OMGYC=SIGN(TRLIM,CMGYC)
90 IF (ABS(CMGZC).GE.TRLIM) OMGZC=SIGN(TRLIM,CMGZC)
   IF (ABS(GIMBAL Y-AXIS ANG RATE TO SEEKER Y-AXIS
   RESOLVE CMGY*CPSSIS+SPSSIS*(R*STHTS-P*CTHTS)
C   SEEKER DYNAMICS
   D2YD=2.4*(CMGYC-OMGYS)
   CMGYC=2.139.*(D2Y+D2YD/64.3)
   D2ZD=4.4*(CMGZC-OMGZ)
   CMGZD=1445.925*(D2Z+D2ZD/64.3)
C   SEEKER EULER ANGLE RATES
   THTS=CMGY-G

```



```

PSISL=CMGZ-P*STHTS-R*CTHTS
THLILIM=SIGN(GMBLIM,THTS)
PSILIM=SIGN(GMBLIM,PSIS)
IF(AES(THTS).GT.GMBLIM) THTSD=50.*(THTLIM-THTS)
IF(ABS(PSIS).GT.GMBLIM) PSISD=50.*(PSILIM-PSIS)
RETURN
200 FORMAT(1H,10X,'END OF SEARCH,NO TARGET DETECTION')
130 FORMAT(1H,10X,'SEEKER LCST TARGET,LCCK ANGLE=',F8.3)
140 FORMAT(1H,10X,'START SEARCH,ATIGS TGT RGE=',F10.3,3X,'T=',F8.3)
150 END

```

CC C C

SUBROUTINE S2I

```

COMMON C(3415)
EQUIVALENCE (C(0336), FLGD)
EQUIVALENCE (C(0347), FLGTS)
FLGD=C.C
FLGTS=C.C
RETURN
END

```

CC CC C

SUBROUTINE S2

RADICMETER MODULE S2

```

COMMON C(3415)
EQUIVALENCE (C(0172), BTA02)
EQUIVALENCE (C(0224), A11)
EQUIVALENCE (C(0228), A12)
EQUIVALENCE (C(0232), A13)
EQUIVALENCE (C(0236), A21)
EQUIVALENCE (C(0240), A22)
EQUIVALENCE (C(0244), A23)
EQUIVALENCE (C(0248), A31)
EQUIVALENCE (C(0252), A32)
EQUIVALENCE (C(0256), A33)
EQUIVALENCE (C(0273), THTS)
EQUIVALENCE (C(0277), PSIS)
EQUIVALENCE (C(0280), ANGLT)
EQUIVALENCE (C(0290), YTP)
EQUIVALENCE (C(0298), ZTP)
EQUIVALENCE (C(0306), XTTP)
EQUIVALENCE (C(0310), YTTTP)
EQUIVALENCE (C(0311), ZTTTP)
EQUIVALENCE (C(0312), ZTTTP)

```



```

EQUUIVALENCE (C0313), XATP)
EQUUIVALENCE (C0314), YATP)
EQUUIVALENCE (C0315), ZATP)
EQUUIVALENCE (C0316), RS)
EQUUIVALENCE (C0317), FLGT)
EQUUIVALENCE (C0318), RMT)
EQUUIVALENCE (C0319), XCT)
EQUUIVALENCE (C0320), YCT)
EQUUIVALENCE (C0321), ZCT)
EQUUIVALENCE (C0335), FLAG)
EQUUIVALENCE (C0336), FLGD)
EQUUIVALENCE (C0339), X2)
EQUUIVALENCE (C0340), Y2)
EQUUIVALENCE (C0341), Z2)
EQUUIVALENCE (C0342), RMA)
EQUUIVALENCE (C0343), XDA)
EQUUIVALENCE (C0344), YDA)
EQUUIVALENCE (C0345), ZDA)
EQUUIVALENCE (C0346), OMGC)
EQUUIVALENCE (C0347), FLGTS)
EQUUIVALENCE (C0348), ERSZ)
EQUUIVALENCE (C0349), ERSZL)
EQUUIVALENCE (C0356), CTHTS)
EQUUIVALENCE (C0357), STHTS)
EQUUIVALENCE (C0358), CPSIS)
EQUUIVALENCE (C0359), T)
EQUUIVALENCE (C0932), TUC)
EQUUIVALENCE (C0946), TUC)
RAC=57.2557795
X1=X1TF-X1TF
Y1=Y1TF-Y1TF
Z1=Z1TF-Z1TF
X2=XATF-X1TF
Y2=YATF-Y1TF
Z2=ZATF-Z1TF

```

C TARGET POSITION IN BODY AXIS SYSTEM

C XMT=A11*X1+A12*Y1+A13*Z1
YMT=A21*X1+A22*Y1+A23*Z1
ZMT=A31*X1+A32*Y1+A33*Z1
RMT=SGRT(XMT**2+YMT**2+ZMT**2)
ATTGS=IGT POSITION IN BODY AXIS SYSTEM

```

XMA=A11*X2+A12*Y2+A13*Z2
YMA=A21*X2+A22*Y2+A23*Z2
ZMA=A31*X2+A32*Y2+A33*Z2
RMA=SGRT(XMA**2+YMA**2+ZMA**2)
IF (T.L.DIRECTION COSINES WRT BODY (PITCH-YAW SEQUENCE)
CTHTS=CCS(THTS)

```

C


```

STHTS=SIN(THTS)
CPSIS=COS(P SIS)
SPSIS=SIN(P SIS)
B11=CPSIS*CTHTS
B12=SPSIS*STHTS
B13=-CPSIS*CTHTS
B21=-SPSIS*CTHTS
B22=CPSIS*STHTS
B31=STHTS
B32=0.
B33=CTHTS
C
  RESOLVE TGT PCSITION FROM BODY SYSTEM TO SEEKER AXES
  XDT=B11*YMT+B12*YMT+B13*ZMT
  YDT=B21*YMT+B22*YMT+B23*ZMT
  ZDT=B31*YMT+B32*YMT+B33*ZMT
  RXYT=SCRT(XDT**2+YDT**2)
  RESOLVE ATIGS TGT POSITION FROM BODY SYSTEM TC SEEKER AXES
  XCA=E11*XMA+B12*YMA+B13*ZMA
  YCA=B21*XMA+B22*YMA+B23*ZMA
  ZCA=B31*XMA+B32*YMA+B33*ZMA
  IF (FLACS.GT.0.0) GC TC
  FLG1=C
  GO TC 20
5
  ANGLTR=ATAN(RYZT/XCT)
  IF (FLGTS.GT.0.0) GC TO 20
  BETA=EFFEC TIVE SEARCH BEAM WIDTH (DEG)
  BTAC2=BETA/2.0
  TEST FCR TARGET DETECTION
  IF (FLG1.GT.0.0) GO TC 10
  FLG1=SET WHEN TARGET ENTERS BEAM
  IF (ABS(ANGLT).LE.BTAC2) FLG1=1.0
  GO TC 20
10
  IF (FLGD.GE.1.0) GC TC 20
  TIME=1
  FLGD=SET WHEN TARGET LEAVES BEAM
  IF (ABS(ANGLT).GE.BTAC2) FLGD=1.0
  GO TC 20
20
  TD=T-TIME
  DETECTION DECLARED 30 MSEC AFTER TGT LEAVES BEAM
  IF (TD.CE.C.03) GO TC 25
  GO TC 20
25
  IF (FLGTS.GE.1.0) GC TC 30
  WRITE(6,160) T
  FLGTS=1.0

```



```

GO TC 200
30 IF (FLGT.LE.0.0) GO TC 200
   RADIOMETER ERROR SIGNAL
   CHI1=2.0193*(2.*ANGLT/3.-1.0)
   CHI2=2.0193*(2.*ANGLT/3.+1.0)
   IF (CHI1.EC.0.0) GO TC 40
   ERSIG=(SIN(CHI1)/CHI1)**2-(SIN(CHI2)/CHI2)**2
   GO TC 60
40 ERSIG=1.C-(SIN(CHI2)/CHI2)**2
   ERSIG=ZERO IF SPEKER LCSES TRACK
50 IF (ANGLE.GT.2.5) ERSIG=0.0
   RESOLVE INTO AZ,EL ERRCR SIGNALS (SMALL ANGLE APPROX)
   IF (RYZT.LE.0.0) GC TC 60
   ERSAZ=ERSIG*(YDT/RYZT)
   ERSSEL=ERSIG*(-ZDT)/RYZT
   GO TC 200
60 ERSAZ=C.0
   ERSSEL=C.0
200 RETURN
160 FORMAT (1F,10X,'END CF SEARCH,TARGET DETECTED.T=',F8.3)
      END
CC
CC
SUERGUTLINE S3I
RETURN
END
CC
SUBROUTINE S3
RETURN
END
CC
SUBROUTINE S4I
RETURN
END
CC
SUBROUTINE S4
RETURN
END
CC
SUBROUTINE S5I
RETURN
END
CC

```


CC
CC
CC

SUBROUTINE S5
ACCELERCMETERS AND GYRCS MCDULE

CCMMCN {(3415)
EQUIVALENCE (C(0136),CG)
EQUIVALENCE (C(0212),PI)
EQUIVALENCE (C(0213),CC)
EQUIVALENCE (C(0216),G)
EQUIVALENCE (C(0217),RC)
EQUIVALENCE (C(0220),R)
EQUIVALENCE (C(0710),AX)
EQUIVALENCE (C(0711),AY)
EQUIVALENCE (C(0712),AZ)
EQUIVALENCE (C(1644),AZM)
EQUIVALENCE (C(1648),AYM)
EQUIVALENCE (C(1652),QM)
EQUIVALENCE (C(1656),RM)
EQUIVALENCE (C(0980),FM)
EQUIVALENCE (C(1749),XAC)

CATA FLGA/O.O/ CFFSET FROM CG
ACCELERCMETERS AT STA. 134.5

XACC=CG-4.5423
AXI=AX-XACC*(R**2+Q**2)

AZI=AZ+XACC*(R*P-QC)

AYI=AY+XACC*(G*P+RC)

IF (FLGA.GT.O.O) GO TO 1C

IDEAL ACCELERCMETERS

AXM=AXI

AYM=AYI

AZM=AZI

GO TO 2

DIGITAL ACCELERMETERS

TRANSFORMATION TO SKEWED ACCELEROMETER AXES

1C EAI1=EAI1

EAI2=EAI1

EAI3=EAI1

EAI1=-EAI1

EAI2=-EAI2

EAI3=-EAI3

EAI1=EAI1

EAI2=EAI2

EAI3=EAI3

EAI1=EAI1

EAI2=EAI2

EAI3=EAI3

AXA=EAI1*AXI+BAI2*AYI+BAI3*AZI

AYA=BAI1*AXI+BAI2*AYI+BAI3*AZI


```

C      AZA=EA31*AX1+BA32*AY1+BA33*AZ1  G
ACCELEFCMETER QUANT. STEP=.194  G
AQS=6.24176
ANSX=AXA/ACS
AXMA=ACS*INT(ANSX)
ANSY=AYA/ACS
AYMA=ACS*INT(ANSY)
ANSZ=AZA/ACS
AZMA=ACS*INT(ANSZ)
TRANSCFORMATION BACK TC ECDY AXES
AXM=BA11*AXMA+BA21*AYMA+BA31*AZMA
AYM=BA12*AXMA+BA22*AYMA+BA32*AZMA
AZM=BA13*AXMA+BA23*AYMA+BA33*AZMA
C      *** ICEAL CYRCS *****
20  GM=C
    RM=R
    FM=P
    RETURN
    END
C      SUBROUTINE S6I
    RETURN
    END
C      SUBROUTINE S6
    RETURN
    END
C      SUBROUTINE S7I
    RETURN
    END
C      SUBROUTINE S7
    RETURN
    END
C      SUBROUTINE S8I
    RETURN
    END
C      SUBROUTINE S8
    RETURN

```



```

C      C
C      ENC
C      SUBRCUTINE S9I
C      RETURN
C      END
C      C
C      SUBRCUTINE S9
C      RETURN
C      ENC
C      C
C      SUBRCUTINE S10I
C      RETURN
C      END
C      C
C      SUBRCUTINE S10
C      RETURN
C      END
C      C
C      SUBRCUTINE BOOST (TI,THRUST,FF)
C      DIMENSION T(25),F(25),W(25)
C      COMMON C(2415)
C      EQUIVALENCE (C(0932),T1)
C      EQUIVALENCE (C(0507),F)
C      DATA T(1)/0.0/,T(2)/.075/,T(3)/.08/,T(4)/.10/,T(5)/.50/,T(6)/1.0/,
1T(7)/1.50/,T(8)/1.75/,T(9)/1.95/,T(10)/2.125/,T(11)/3.125/,
2T(12)/3.505/,T(13)/3.70/,T(14)/3.80/,T(15)/3.875/,T(16)/3.95/,
3T(17)/4.025/,T(18)/4.075/,T(19)/4.15/,T(20)/4.20/,T(21)/4.25/,
4T(22)/4.35/,T(23)/4.475/,T(24)/4.60/,T(25)/4.75/
C      DATA F(1)/0.0/,F(2)/2000./,F(3)/16000./,F(4)/17800./,F(5)/19200./,
1F(6)/21500./,F(7)/24100./,F(8)/25100./,F(9)/25800./,F(10)/26100./,
2F(11)/26500./,F(12)/26500./,F(13)/26400./,F(14)/25600./,
3F(15)/24800./,F(16)/24000./,F(17)/22000./,F(18)/16000./,
4F(19)/12000./,F(20)/5000./,F(21)/6900./,F(22)/4000./,F(23)/1900./,
5F(24)/500./,F(25)/0.C/
C      DATA W(1)/C.C/,W(2)/.321/,W(3)/2.841/,W(4)/6.575/,W(5)/61.178/,
1W(6)/111.759/,W(7)/148.870/,W(8)/172.301/,W(9)/182.143/,
2W(10)/206.751/,W(11)/255.867/,W(12)/362.863/,W(13)/376.027/,
3W(14)/581.912/,W(15)/388.351/,W(16)/392.396/,W(17)/397.734/,

```



```

4W(18)/401.847/,W(19)/404.344/,W(20)/406.206/,W(21)/407.268/,
5W(22)/408.396/,W(23)/409.175/,W(24)/409.883/,W(25)/410.296/

```

```

TIME=TI-TI
IF(TIME.GT.C.) GO TC 3
FF=0.0
THRUST = C.0
GO TC 4
DO 1 I = 2,25
IF(T(I).GE.TIME) GC TC 2
CONTINUE
CALL EXIT
2 AK=(TIME-T(I-1))/(T(I)-T(I-1))
THRUST = F(I-1) + (F(I) - F(I-1))*AK
FF = W(I-1) + (W(I) - W(I-1))*AK
PSF=2116.217*(1.C-(6.87534703E-06)*H)**5.25627005)
THRUST=THRUST+(2116.2-PSF)*.715428
RETLN
END

```

```

SUBROUTINE RAMJET (FIN,XMIN,AIN,WFIN,XIN,CT)

```

```

SIMPLIFIED RAMJET MODEL
USES TABLE LOOK-UP FOR THRUST COEFF AND FUEL FLOW RATE

```

```

DIMENSION TM1(6),TM2(7),TH(4),TETA(5),TCF1(6,5,4),TCF2(7,5,4),
1TW(6,5,4)
COMMON C(2415)

```

```

DATA TM1/2.3,2.5,2.7,2.9,3.1,3.3/

```

```

DATA TM2/C.5,1.0,1.5,2.0,2.5,3.0,3.3/

```

```

DATA TF/500.,13000.,20000.,35000./

```

```

DATA TETA/C.,2.,4.,6.,8./

```

```

DATA TCF1/ .3814,.2958,.2075,.1374,
A.0844,.0443,.3794,.2843,.2025,.1339,.0805,.0402,.3731,.2818,
B.1915,.1203,.0666,.0258,.3476,.2526,.1659,.1049,.0555,.0173,
C.3251,.2355,.1595,.1001,.0549,.0291,.4357,.3412,.2453,.1651,
D.1115,.0682,.4337,.3293,.2401,.1654,.1075,.0639,.4273,.3268,
E.2287,.1512,.0927,.0485,.4012,.2967,.2061,.1350,.0810,.0396,
F.3782,.2751,.1954,.1300,.0805,.0425,.4944,.386,.2798,.1963,
G.1341,.088,.4924,.373,.2742,.1924,.1256,.0832,.4857,.3706,.2618,
H.1766,.1132,.066,.4573,.3376,.237,.1588,.1003,.056,.4328,.3186,

```


I 2254, 1535, 0999, 6273, 5139, 3561, 2997, 2243, 1661, 2054,
 J 625, 5028, 3911, 2202, 1616, 6175, 4598, 3801, 2821, 2054,
 K 1455, 5531, 4721, 3588, 2662, 1936, 1367, 5708, 4555, 3484,
 L 2613, 1532, 1398/

C

DATA TCF2/ -5727, -3661, -4238, 5727, -3658, -4235, -2618, -2067,
 A-2615, -2067, -2074, -2091, -4233, -2623, -2065, -2069, -2085,
 B-2072, -209, -5728, -3655, -4233, -2623, -2065, -2069, -2085,
 C-5725, -2631, -4218, -2635, -2081, -2064, -2065, -2069, -2085,
 D-4205, -2648, -2062, -2061, -2081, -2064, -2065, -2069, -2085,
 E-2074, -2060, -2075, -2061, -2081, -2064, -2065, -2069, -2085,
 F-2074, -2060, -2075, -2061, -2081, -2064, -2065, -2069, -2085,
 G-3619, -4207, -2624, -2061, -2081, -2064, -2065, -2069, -2085,
 H-2636, -2054, -2074, -2061, -2081, -2064, -2065, -2069, -2085,
 I-2041, -2054, -2074, -2061, -2081, -2064, -2065, -2069, -2085,
 J-5666, -2054, -2074, -2061, -2081, -2064, -2065, -2069, -2085,
 K-4154, -2054, -2074, -2061, -2081, -2064, -2065, -2069, -2085,
 L-2035, -2054, -2074, -2061, -2081, -2064, -2065, -2069, -2085,
 M-2043, -2054, -2074, -2061, -2081, -2064, -2065, -2069, -2085,
 N-3556, -2054, -2074, -2061, -2081, -2064, -2065, -2069, -2085,
 O-2589, -2054, -2074, -2061, -2081, -2064, -2065, -2069, -2085,
 P-2017, -2054, -2074, -2061, -2081, -2064, -2065, -2069, -2085/

C

DATA TWF/ 2.95, 2.95, 2.88, 2.83, 2.83, 2.83, 2.83, 2.83, 2.83,
 A2.806, 2.806, 2.806, 2.806, 2.806, 2.806, 2.806, 2.806, 2.806,
 B2.775, 2.775, 2.775, 2.775, 2.775, 2.775, 2.775, 2.775, 2.775,
 C2.604, 2.604, 2.604, 2.604, 2.604, 2.604, 2.604, 2.604, 2.604,
 D1.876, 1.876, 1.876, 1.876, 1.876, 1.876, 1.876, 1.876, 1.876,
 E1.812, 1.812, 1.812, 1.812, 1.812, 1.812, 1.812, 1.812, 1.812,
 F1.73, 1.73, 1.73, 1.73, 1.73, 1.73, 1.73, 1.73, 1.73,
 G1.557, 1.557, 1.557, 1.557, 1.557, 1.557, 1.557, 1.557, 1.557,
 H1.42, 1.42, 1.42, 1.42, 1.42, 1.42, 1.42, 1.42, 1.42,
 I1.373, 1.373, 1.373, 1.373, 1.373, 1.373, 1.373, 1.373, 1.373,
 J.933, .933, .933, .933, .933, .933, .933, .933, .933,
 K.872, .872, .872, .872, .872, .872, .872, .872, .872/

C

F=HIN*3.280829895
 IF (XIN.GT.0) GO TC 10
 R (XIN.GT.0) CRUISE
 CT=TH-RECL(XMIN,AIN,H,TM1,TETA,TH,TCF1,6,5,4)
 WFIN=TH-RECL(XMIN,AIN,H,TM1,TETA,TH,TCF1,6,5,4)
 GO TC 20
 RAMJET-CT=TH-RECL(XMIN,AIN,H,TM2,TETA,TH,TCF2,7,5,4)
 10 CT=TH-RECL(XMIN,AIN,H,TM2,TETA,TH,TCF2,7,5,4)
 WFIN=0.0
 RETURN
 20 END

C

1 SUERCUTINE ENGINE (FIN,XMIN,AIN,WFIN,TIN,XIN,YIN,ZIN,CT,SMARG,
IFI,IFREAD)

NWC COMMON DECK FOR AIRBREATHING PROPULSION PROGRAMS
CPIA NCMENCLATURE

REAL ISF,ISTAR,M,M1,M2,M3,M4,M5,M6,MW,MW1,MW2,MW3,MW4,MW5,MW6
REAL MACR,MACRS,MACF
INTEGER CCNT

COMMON/CRUS/ ENG(30),IENG(15)
COMMON/RJ/ A,AINF,A1,A2,A3,A4,A5,A6,AR,AFR,AFS,ALPHA
COMMON/RJ/ ACR,BLEED
COMMON/RJ/ CDASUB,CCASUP,CDB,CF,CFINF,CF6,CFB,CFT,CFC,CNM
COMMON/RJ/ ER,ERRL,ETAC,ETAN,F
COMMON/RJ/ FAR,G,GAMMA,GAMMA1,GAMMA2,GAMMA3,GAMMA4,GAMMA5,GAMMA6
COMMON/RJ/ H,ISP,ISTAR,I1,I2,I3,I4,I5,I6,I7,I8,I9,I10
COMMON/RJ/ CCNT,ICCNV,ICCNVP,IFIRST
COMMON/RJ/ M,M1,M2,M3,M4,M5,M6,MW,MW1,MW2,MW3,MW4,MW5,MW6
COMMON/RJ/ MACR,MACRS,PM
COMMON/RJ/ P,P1,P2,P3,P4,P5,P6,PT30I,PRAR
COMMON/RJ/ PT,PT1,PT2,PT3,PT4,PT5,PT6,PMOPT,PMOPT1,PMOPT2,PMOPT3
COMMON/RJ/ PHI1,PHI2,PHI3,PHI4,PHI5,PHI6,PHIPRI,Q
COMMON/RJ/ RADEG,R,SA,SART,SQPTNT
COMMON/RJ/ T,T1,T2,T3,T4,T5,T6,TIME
COMMON/RJ/ TT,TT1,TT2,TT3,TT4,TT5,TT6
COMMON/RJ/ V,V1,V2,V3,V4,V5,V6
COMMON/RJ/ WA,WF,X,Y,Z

* * * * *

COMMON/EXFALS/ TABLE(1000)

COMMON/WRITCT/ WRIT(20)

NAMelist /MCNT/ ENG,IENG

*** THE INFORMATION BELOW NOTES HOW THE NAMELIST
DATA IS INPUT AND WHAT THE INPUTS ARE ***

ENG(1)=A1/AR OR A1 (M SC) SEE I6
ENG(2)=A3/AR OR A3 (M SC) SEE I6
ENG(3)=A4/AR OR A4 (M SC) SEE I6
ENG(4)=A5/AR OR A5 (M SC) SEE I6


```

ENG(5)=A6/AR CR A6 (M SQ) SEE I6
ENG(6)=AR (METRES SQUARED)
ENG(7)=CCE BURNER DRAG COEFFICIENT
ENG(8)=ETAC, NOZZLE THRUST EFFICIENCY
ENG(9)=ETAC, COMBUSTION EFFICIENCY
ENG(10)=WF, FUEL FLOW RATE (KG/SEC) SEE I5
ENG(11)=ISP, STAR (FOR GAS GEN SYSTEMS)
ENG(12)=PFI, CF PRIMARY (FOR GAS GEN SYSTEMS)
ENG(13)=ELEC, FLOW PERCENTAGE/100
ENG(14)=AIR, STICICHEMETRIC (AFS)
ENG(15)=CN, NOZZLE MASS FLOW COEFFICIENT

REAL MCCSE MCCSE(176), EAGLE(176)
DIMENSION MCCSE(176), EAGLE(176)

***
THE INFLTS BELOW ARE SWITCHES ***
IENG(1)=I1, ENGINE TYPE SWITCH
IF I1=0, RAMJET TYPE IF I1=1, GAS GEN TYPE
IENG(2)=I2, COMBUSTION EFF SWITCH
IF I2=C, ETAC IS SET BY INPUT
IF I2=1, ETAC IS SET IN A SUBROUTINE
IENG(3)=I3, ATMOSPHERE CHOICE SELECTION, SEE MEMO BY RGL
IENG(4)=I4, STANDARD DAY SWITCH
IF I4=C, REAL AIR SOLUTION
IF I4=1, USE THE GAMMA=1.4 SOLUTION
IENG(5)=I5, WF OR SOLUTION SWITCH
IF I5=1, THE PROGRAM USES WF IN THE SOLUTION
IF I5=2, THE PROGRAM USES WF IN THE SOLUTION
IENG(6)=I6, INLET CATS TYPE SWITCH
IF I6=C, THE INPLTS ARE RATIOS BASED ON AR
IF I6=1, THE AREAS ARE ABSOLUTE VALUES
IENG(7)=I7, PRINT EACH ITERATION SWITCH
IF I7=0, DO NOT PRINT ITERATION
IF I7=1, DO PRINT ITERATION
IENG(8)=I8, COMBUSTION PRESSURE EFFECT SWITCH
IF I8=0, THERE IS NO PRESSURE EFFECT COMPUTED
IF I8=1, THE PRESSURE EFFECT ON COMBUSTION TEMP IS COMPUTED

EQUIVALENCE(TABLE(206),MCCSE(1))
EQUIVALENCE(TABLE(382),EAGLE(1))

***
EXAMPLE CF NAMELIST INPUT ***
$MCNT
ENG=.5256,.85,.85,.44444,.56,.1140092,0,.58,.55,2,.50,1.2,.04,13.8
1.0,
IENG=C,0,2,C,1,0,0,0,
IEND

```



```

C C C C C C C C C C
I3=IENG(3)
I4=IENG(4)
I5=IENG(5)
I6=IENG(6)
I7=IENG(7)
I8=IENG(8)
IF ( I6.EC.O ) A1=A1*AR
IF ( I6.EC.O ) A3=A3*AR
IF ( I6.EC.C ) A4=A4*AF
IF ( I6.EC.O ) A5X=A5*AR
IF ( I6.EC.O ) A6=A6*AF
IF ( I6.EC.1 ) A5X=A5
A5=A5*CNM
IF ( IFIRST.EC.-1 ) GC TC 20
GO TC 30
C COMMENT INITIALIZE , SET CONSTANTS AND WRITE AREAS
20 CONTINUE
AIRBREATHING GEOMETRY ONLY CALLED ON FIRST PASS
CALCULATE GEOMETRY PARAMETERS
A4OA5=A4/A5
GAM=S.O/T.O
N4=FMAFL(GAM,A4OA5)
PHI4=PTIN(GAM,M4)
A6CA5=A6/A5
N6=FMAFH(GAM,A6OA5)
PHI6=PTIN(GAM,M6)
IDENTIFY AND PRINT PARAMETERS
WRITE (6,420) A1
WRITE (6,430) A3
WRITE (6,440) A4
WRITE (6,450) A5X
WRITE (6,460) A6
WRITE (6,470) AR
WRITE (6,480) PHI4
WRITE (6,490) PHI6
WRITE (6,500) CDR
WRITE (6,510) CNM
WRITE (6,520) CNM
20 CONTINUE
SET CONSTANTS AND INITIALIZE SUBROUTINES
G IN KG/M/SEC SQ - N
C C C C C C C C C C

```



```

C      RADEG=.572557779E+02
C      G=9.80665
C      START CALCULATIONS FOR EACH NEW POINT
C      COUNT=0
C      ICCNVP=1
C      NEW ITERATIONS START HERE
C      4C ICCNV=C
C      CHECK TC SEE IF 100 ITERATIONS OF THE SAME PCINT HAVE BEEN MADE
C      IF (CCUNT-100) 60,60,50
C      IF NCT CCNVERGED AFTER 100 ATTEMPTS, STOP
C      50 WRITE (6,530)
C      CALL EXIT
C      IF NCT CCNVERGED WITH LESS THAN 100 ATTEMPTS, TRY AGAIN
C      6C CONTINUE
C      IF (CCUNT-CT.89) I7=1C
C      IF AECVE STATEMENT NCT CCNVERG,PRINT LAST 10
C      IF (CCUNT-NE.0) GO TC 50
C      IF (IFIRST.EC.-1) CALL AIR(H,M,-1,P,T,A,DEN,Q,G,1,1)
C      IF (IFIRST.EC.-1) GO TC 70
C      I3=I+10C
C      CALL AIR(H,M,1,P,T,A,DEN,Q,G,1,1)
C      G=9.80665
C      IF (I4.EC.1) GO TO 8C
C      ENTHR=46.4278+.23945*(T-194.444)
C      V IN METRES PER SEC
C      V=M*A
C      ENRG IN CAL/GM
C      ENRG=(V**2.)/8368.
C      ENTHAL=ENTHR+ENRG
C      ECGM IN CAL/GM
C      ECGM=ENTHAL+ENRG
C      TT3 IN DEGREE RANKINE
C      TT3=539.31+(7.5164*E(CGM))-(3.3559E-03*ECGM*ECGM)+(4.65976E-06*(ECGM
C      1**3.))-((1.32956E-09*(ECGM**4.))
C      TT3=TT3/1.8
C      GAMMA3=1.4048-(6.09654E-04*ECGM)+(2.12102E-06*ECGM*ECGM)-(5.56849E

```



```

1-QS*(ECGM**2.)))+(6.49543E-12*(ECGM**4.))
GO TC 50
CONTINUE
EC GAMMA3=1.4
TT3=1/FT(TT(1.4,M))
T3=TT3*FICTT(1.4,M3)
SC CONTINUE
C
IF (IFIRST) 100,11C,11C
100 WRITE (6,540)
C
COMBUSTCR
C
IF (I2.EG.1) CALL ETA
CALL TAE (AFS)
V=-0.0
WA=1C.C
M3=.2
MACRS=1.
CALL PCRA
C
CALCULATION
C
WRITE (6,230)
C
DIFFUSER
C
CALL HCF
GO TC 15C
CONTINUE
11C
IF (M.GT.1.07) GO TC 130
*** BEGIN SUBSONIC CALL SEQUENCE *****
IF (WF.LT..001) GO TC 120
PT=P/FFCPT(1.4,M)
PMOPT=FFCPT(1.4,M)
TT=T/FTTT(1.4,M)
ACR=.27861+2.4721*M-4.527*M*M+2.3036*M*M*M
AINF=A1*ACR*1.05
WA=(AINF*FT*PMOPT)/SCRT(TT)
CALL HCRA
CONTINUE
120
IF (WF.LT..001) ER=0.0
FAR=ER/AFS
CALL SLESCN (H,M,FAR,CF,WF,Q,AR)
ICCNV=C
COUNT=C
GO TC 225
*** END OF SUBSONIC CALL SEQUENCE ***
C

```



```

C 13C CONTINUE
C
C ***** WE SET SA HERE *****
IF (15.EG.1) GO TO 140
IF (15.EG.2) GO TO 150
WRITE (6,240)
C GC THIS ROUTE IF ER IS INPUT OR CHANGED IN HCRA
140 CONTINUE
CALL HCRA
IF (IFUEL.EG.5) ER=0.0
FAR=ER/AFS
WF=FAR*WA
AFR=1./FAR
GO TC 160
C GC THIS ROUTE IS WF IS INPUT OR SET IN HCRA
150 CONTINUE
CALL HCRA
IF (IFUEL.EG.5) WF=0.0
FAR=WF/WA
ER=FAR*AFS
IF (FAR.EG.0.0) GO TC 1400
AFR=1./FAR
GO TC 160
1400 AFR=0.0
C 16C CONTINUE
C
TOTT=F7CTT(GAMMA3,M3)
T3=TT3*TC*TT3
SQRTCT=SQRT(TT3)
C TO FIND AIR SPECIFIC IMPULSE
C
IF (110.EG.0) GO TO 161
CONTINUE
GO TC 165
161 CONTINUE
CALL INTR20 (TT3,ER,TT41,TABLE,1)
CALL INTR20 (TT3,ER,GAMMA4,TABLE,2)
CALL INTR20 (TT3,ER,NW4,TABLE,3)
C COMPUTE THE EFFECT OF PRESSURE ON TEMPERATURE
C
IF (18.EG.0) GO TO 162
G1=GAMMA4+1.0
G2=GAMMA4-1.0
G3=G1/G2

```



```

G4=2.*G1*((2./G1)**G3)
G5=SQRT(G4)
PT4=(SA*WA)/(G5*A5)
PT4ATN=PT4/101360.
IF(PT4ATN.LT.2) PT4ATN=.2
IF(PT4ATN.GT.20.) PT4ATN=20.
IF(113.LT.255.) GC TC 1615
IF(ER.GT.1.5) GO TO 1615
CTEMP=TC(TT3,ER,PT4ATN)
CONTINUE
TT4I=TT4I+CTEMP
1615 CCNTINLE
162 CCNTINLE
C
C
C NCW FINISH COMBUSTOR COMPUTATIONS
R=8314.24/MW4
WAX=WA/.4525924
THROAT=A5X/.C9290304
TTT=((TT3*1.8)/1000.)*2.
RSV=(WA)*TTT)/THROAT
IF(I2.EG.1) CALL ETA
TT4=ETAC*(TT4I-TT3)+TT3
BBB=2.*C*R*(1.0+GAMMA4)*TT4/GAMMA4
BBB=AMAX1(400.,BBB)
SA=(1.0+FAF)*SQRT(BEE)
SART=SA/SQRTCT
CONTINUE
165 C
C
C COMMENT MAKE CALCULATIONS TAKING VARIABLE NOZZLE GAMMA INTO EFFECT
A4CA5=A4/A5
M4=FMARL(GAMMA4,A4CA5)
PTI4=PTIN(GAMMA4,M4)
C
C
C ***** WE COMPUTE CONDITIONS AT STATION 3 HERE *****
STRCG=22.5612
IF(11.EQ.1) GO TO 170
RAMJET COMPUTATIONS - FUEL ADDED IN THE INLETS
SARTA=STFCG*SQRT((GAMMA3+1.)/GAMMA3)*(1.+FAF)
PTI3=(PTI4*SART/SARTA)+((GAMMA3*CDB*M3*M3)/(2.*XMCIR3*SARTA))
GO TC 180
GAS GENERATOR SYSTEM COMPUTATIONS
SARTE=STFCG*SQRT((GAMMA3+1.)/GAMMA3)
PHI3=(PTI4*SART/SARTE)-((PHIPRI*ISTAR*FAF)/(SARTB*SQRTGT))+((GAMMA
13*CDB*M3*M3)/(2.*XMCIR3*SARTE))
CONTINUE
180 N3=FNPHIN(GAMMA3,PHI3)

```



```

C C C FMCFT3=FFMFCFT(GAMMA3,M3)
C C C PMOFT1=FFMFCFT(1,4,M)
C C C XMCIF3=FFMFCIR(GAMMA3,M3)
C C C ***** WE NOW CALL THE INLET ROUTINE TO GET WA *****
C C C CALL HCF
C C C 190 CONTINUE
C C C ***** DEBUG OUTPUT *****
C C C IF (I7.EC.0) GO TC 200
C C C 201 CONTINUE
C C C FK=F/1000
C C C FLK=F1/1000
C C C F2K=F2/1000
C C C P3K=P3/1000
C C C P4K=P4/1000
C C C F5K=F5/1000
C C C P6K=P6/1000
C C C FTK=FT/1000
C C C FT1K=FT1/1000
C C C PT2K=PT2/1000
C C C FT3K=PT3/1000
C C C PT4K=PT4/1000
C C C PT5K=PT5/1000
C C C PT6K=PT6/1000
C C C GK=G/1000
C C C WRITE (6,250) A,AINF,A1,A2,A3,A4,A5,A6,AR,AFR,AFS,ALPHA
C C C WRITE (6,260) ACR,BLEED
C C C WRITE (6,270) CDASUB,CDASUP,CDB,CF,CFINF,CF6,CFB,CFT,CFC,CNM
C C C WRITE (6,280) ER,ERRL,ERRLL,ETAC,ETAN,F
C C C WRITE (6,290) FAR,G,GAMMA,GAMMA1,GAMMA2,GAMMA3,GAMMA4,GAMMA5,GAMMA
16 WRITE (6,300) H,ISP,ISTAR,I1,I2,I3,I4,I5,I6,I7,I8,I9,I10
C C C WRITE (6,310) COUNT,ICCNV,ICONVP,IFIRST
C C C WRITE (6,320) M,M1,M2,M3,M4,M5,M6,MW,Mh1,Mh2,Mh3,Mh4,Mh5,Mh6
C C C WRITE (6,330) MACP,MACRS,PM
C C C WRITE (6,340) PK,P1K,P2K,P3K,P4K,P5K,P6K,PT3CI,PRAR
C C C WRITE (6,350) PTK,PT1K,PT2K,PT3K,PT4K,PT5K,PT6K,PMOFT,PMOFT1,PMOFT
12,PMOFT1
C C C WRITE (6,360) PHI1,PHI2,PHI3,PHI4,PHI5,PHI6,PHIPRI,QK
C C C WRITE (6,370) RADEG,F,SA,SART,SQRTOT
C C C WRITE (6,380) T,T1,T2,T3,T4,T5,T6,TIME
C C C WRITE (6,390) TT,TT1,TT2,TT3,TT4,TT5,TT6
C C C WRITE (6,400) V,V1,V2,V3,V4,V5,V6
C C C WRITE (6,410) WA,WF,X,Y,Z

```



```

C 20C CONTINUE, INCREASE CCUNT AND CALCULATE PARAMETERS NOT IN COMMON
C SET IFTS,
C COUNT=CCUNT+1
C CHECK CONVERGENCE, ICCNVF IS USED TO MAKE SOLUTION CONVERGE TWICE
C IF CCNVERGED, PRINT. IF NOT, GO BACK AND TRY AGAIN
C
C 21C IF (IFIRST.EC.-1) GO TO 225
C ICCNVF=ICCNV
C GO TO 4C
C 22C CONTINUE
C ***** WE NOW COMPUTE THRUST IF CCNVERGED *****
C
C 221 A6CA5=A6/A5
C M6=FMARH(GAMMA4,A6CA5)
C PHI6=PHIN(GAMMA4,M6)
C IF(110.*GE.5) GO TO 221
C CF6=(WA*SA*PHI6*ETAN-A6*F)/(AR*Q)
C CONTINUE
C CFINF=2.*ACR*MACRS*A1/AR
C CALPHA=CCS(ALPHA/RADEG)
C CF=CF6-(CFINF*CALPHA)+CFB+CFC
C F=CF*G*AR
C IF(WF.EC.0.0) GO TO 1200
C ISP=F/(WF*G)
C GO TO 1500
C 130C ISP=C.C
C 150C CONTINUE
C G1=GAMMA4+1.0
C G2=GAMMA4-1.0
C G3=G1/G2
C G4=2.*G1*((2./G1)**G3)
C G5=SCRT(G4)
C PT5=(SA*WA)/(G5*A5)
C
C PREPARE DATA FOR TRANSFER TO MAIN PROGRAM
C
C 225 CONTINUE
C FIN=F
C XMIN=M
C AIN=ALFFA
C WFIN=WF
C CDADC=CCASUB+CDA SUP+CFT
C CT=CF-CCACC

```



```

SMARG=PM
IFI=IFIRST
FCCUNT=CCUNT+.01
WRIT(1)=PT3CI
WRIT(2)=WA
WRIT(3)=TT4
WRIT(4)=MACR
WRIT(5)=ACR
WRIT(6)=ER
WRIT(7)=ISP
WRIT(8)=FCCUNT
WRIT(9)=WF
WRIT(10)=ETAC
WRIT(11)=CT
WRIT(12)=FM
WRIT(13)=PCCUNT
WRIT(14)=ESV
IF(PT3.EC.C) GO TC 120C
WRIT(15)=PT5/PT3
120C CONTINUE
WRIT(16)=PT5/1000.
FCCUNT=FCCUNT
RETURN

```

C

```

2300 FORMAT (1, F8.2, 8F7.4, 2F7.3, F7.2)
2400 FORMAT (1, F8.2, 8F7.4, 2F7.3, F7.2)
2500 FORMAT (1, F8.2, 8F7.4, 2F7.3, F7.2)
2600 FORMAT (1, F8.2, 8F7.4, 2F7.3, F7.2)
2700 FORMAT (1, F8.2, 8F7.4, 2F7.3, F7.2)
2800 FORMAT (1, F8.2, 8F7.4, 2F7.3, F7.2)
2900 FORMAT (1, F8.2, 8F7.4, 2F7.3, F7.2)
3000 FORMAT (1, F8.2, 8F7.4, 2F7.3, F7.2)
3100 FORMAT (1, F8.2, 8F7.4, 2F7.3, F7.2)
3200 FORMAT (1, F8.2, 8F7.4, 2F7.3, F7.2)
3300 FORMAT (1, F8.2, 8F7.4, 2F7.3, F7.2)
3400 FORMAT (1, F8.2, 8F7.4, 2F7.3, F7.2)
3500 FORMAT (1, F8.2, 8F7.4, 2F7.3, F7.2)
3600 FORMAT (1, F8.2, 8F7.4, 2F7.3, F7.2)
3700 FORMAT (1, F8.2, 8F7.4, 2F7.3, F7.2)
3800 FORMAT (1, F8.2, 8F7.4, 2F7.3, F7.2)
3900 FORMAT (1, F8.2, 8F7.4, 2F7.3, F7.2)
4000 FORMAT (1, F8.2, 8F7.4, 2F7.3, F7.2)
4100 FORMAT (1, F8.2, 8F7.4, 2F7.3, F7.2)
4200 FORMAT (1, F8.2, 8F7.4, 2F7.3, F7.2)
4300 FORMAT (1, F8.2, 8F7.4, 2F7.3, F7.2)
4400 FORMAT (1, F8.2, 8F7.4, 2F7.3, F7.2)
4500 FORMAT (1, F8.2, 8F7.4, 2F7.3, F7.2)
4600 FORMAT (1, F8.2, 8F7.4, 2F7.3, F7.2)

```



```

470 FCRMAT (14H      A 6 = F9.7,9H SQ METRE)
480 FCRMAT (14H      A R = F9.7,9H SQ METRE)
490 FCRMAT (28H      PFI4 = F7.5)
500 FCRMAT (28H      PFI6 = F7.5)
510 FCRMAT (28H      BURNER DRAG COEFF. = F6.2)
520 FCRMAT (28H      NCZLE MASS COEFF. = F6.2)
530 FCRMAT (36H      DID NOT CONVERGE AFTER 100 ATTEMPTS)
540 FCRMAT (47H      RAMJET PROPULSION
)

END

SUBROUTINE ETA

NWC COMMON DECK FOR AIRBREATHING PROPULSION PROGRAMS
CPIA NOMENCLATURE

REAL ISF,ISTAR,M,M1,M2,M3,M4,M5,M6,MW,MW1,MW2,MW3,MW4,MW5,MW6
INTEGER CCUNT

COMMON /RJ/ A,AINF,A1,A2,A3,A4,A5,A6,AR,AFR,AFS,ALPHA
COMMON /RJ/ ACR,BLFEI
COMMON /RJ/ CDASUB,CDASUF,CDB,CF,CFINF,CF6,CFB,CFT,CFC,CNM
COMMON /RJ/ ERLL,ETAC,ETAN,F
COMMON /RJ/ FAR,G,GAMMA,GAMMA1,GAMMA2,GAMMA3,GAMMA4,GAMMA5,GAMMA6
COMMON /RJ/ F,ISP,ISTAR,I1,I2,I3,I4,I5,I6,I7,I8,I9,I10
COMMON /RJ/ CCUNT,ICCNV,ICCONVP,IFIRST
COMMON /RJ/ M,M1,M2,M3,M4,M5,M6,MW,MW1,MW2,MW3,MW4,MW5,MW6
COMMON /RJ/ MACR,MACRS,PM
COMMON /RJ/ P,P1,P2,P3,P4,P5,P6,PT30I,PRAR
COMMON /RJ/ PT,PT1,PT2,PT3,PT4,PT5,PT6,PMOPT,PMOPT1,PMOPT2,PMOPT3
COMMON /RJ/ PH1,PH2,PH3,PH4,PH5,PH6,PHI1,PHI2,PHI3,PHI4,PHI5,PHI6,PHIPRI,Q
COMMON /RJ/ RADEG,R,SA,SART,SQRTOT
COMMON /RJ/ T,T1,T2,T3,T4,T5,T6,TIME
COMMON /RJ/ TT,TT1,TT2,TT3,TT4,TT5,TT6
COMMON /RJ/ V,V1,V2,V3,V4,V5,V6
COMMON /RJ/ WA,WFX,X,Y,Z

DATA ALL,ELL,C11,C11,F11,F11,G11
* /-14C7.66, .95416, -8.55033, 43.7266, 1468.91, 80484.,
* -277C56./

FAR2=FAR*FAR
FOVER=1./(1.+FAR)

```



```

TERM1=B11+C11*FAR+D11*FAF2
TERM2=E11+F11*FAR+G11*FAR2
TTR=1.E*TT
TT4I=A11+TTR*TERM1+FCVER*TERM2
STT4I=SGRT(TT4I)
ETAC=40.246*STT4I-0.3513*TT4I-1055.08
ETAC=ETAC/1CC.
IF(ER.LT. .0001) ETAC=0.0
RETURN
END

SUBROUTINE PCRA
DIMENSION PS(66), PSTAT1(11), FUELF(11)

NWC COMMON DECK FOR AIRBREATHING PROPULSION PROGRAMS
CPIA NCMENCATURE

REAL ISP,ISTAR,M,M1,M2,M3,M4,M5,M6,MW,MW1,MW2,MW3,MW4,MW5,MW6
REAL MACR,MACRS
INTEGER CCUNT

COMMON /RJ/ A,AINF,A1,A2,A3,A4,A5,A6,AR,AFR,AFS,ALPHA
COMMON /RJ/ ACR,BLEEC
COMMON /RJ/ CDA,SUB,CDA,SUF,CDB,CF,CFINF,CF6,CFB,CFT,CFC,CNM
COMMON /RJ/ ER,ERRL,ERLL,ETAC,ETAN,F
COMMON /RJ/ FAR,G,GAMMA,GAMMA1,GAMMA2,GAMMA3,GAMMA4,GAMMA5,GAMMA6
COMMON /RJ/ F,ISP,ISTAR,I1,I2,I3,I4,I5,I6,I7,I8,I9,I10
COMMON /RJ/ CCUNT,ICCNV,ICCNVP,IFIRST
COMMON /RJ/ M,M1,M2,M3,M4,M5,M6,MW,MW1,MW2,MW3,MW4,MW5,MW6
COMMON /RJ/ MACR,MACRS,PM
COMMON /RJ/ P,P1,P2,P3,P4,P5,P6,PT30I,PRAR
COMMON /RJ/ PT,PT1,PT2,PT3,PT4,PT5,PT6,PMOPT,PMOPT1,PMOPT2,PMOPT3
COMMON /RJ/ PH1,PH12,PH13,PH14,PH15,PH16,PHIPRI,Q
COMMON /RJ/ RADEG,R,SA,SAF,SAFOT
COMMON /RJ/ T,T1,T2,T3,T4,T5,T6,TIME
COMMON /RJ/ TT,TT1,TT2,TT3,TT4,TT5,TT6
COMMON /RJ/ V,V1,V2,V3,V4,V5,V6
COMMON /RJ/ WA,WF,X,Y,Z

ANGLE CF ATTACK PS(I) I=3,14
MACH NUMBER PS(I) I=15,81
DATA PS/

```

CC C C CC CC CC C

CC CC CC C


```

A12.0,4,C,C,0,0,5,1,0,1,5,2,0,
A2.5,3,C,4,C,5,0,6,C,7,C,8,0,
A1.58407,93504,89727,88386,
A.58069,93504,89727,88386,
A.57828,93504,89689,88275,
A.57586,93410,89463,88165,
A.57345,93295,89313,87833,
A.57056,93039,89087,87501,
A.56815,92808,88711,87058,
A.56520,92158,87733,85843,
A.55704,91044,86076,84183,
A.54068,89227,83781,81749,
A.54111,87331,81185,78210,
A.52905,85267,79190,75666/

C      DATA FSTAT1/3.07,3.9,4.9,6.02,7.8.,9.1,10.,11.,12.,13.18/
C
C      DATA FUELF/.94,1.130,1.375,1.628,1.859,
C      A 2.056,2.250,2.416,2.562,2.722,3.05/

      IF (IFIRST) 10,20,20
10 CONTINUE
   WRITE (6,50)
   RETURN
20 CONTINUE
   IF (COUNT-1) 30,30,40
30 CONTINUE
   CALL INTR2C (ALPHA,M,PS1CP,PS,1)
   PSI=PS1CP*(P/6894.757)
   TFFPS=CDLI(PS1,PSTAT1,FUELF,11)
   TFF=TFFPS*.4535924
   WF=TFF
40 RETURN

C 50 FORMAT (T6,'FUEL CONTROL FOR STV G  ')
C
C      END
C
C      SUBROUTINE PCF
C      DIMENSION PR(81), AA(81), CADD(12), XM(12), BWR(12)
C      NWC COMMON DECK FOR AIRBREATHING PROPULSION PROGRAMS
C      CPIA NCMENCLATURE

```



```

REAL ISF,ISTAR,M,M1,M2,M3,M4,M5,M6,MW,MW1,MW2,MW3,MW4,MW5,MW6
REAL MACR,MACRS
INTEGER CCNT

CCMCMCN /RJ/ A,AINF,A1,A2,A3,A4,A5,A6,AR,AFR,AFS,ALPHA
CCMCMCN /RJ/ ACR,BLEED
CCMCMCN /RJ/ CDASUB,CCASUP,CDB,CF,CFINF,CF6,CFB,CFT,CFC,CNM
CCMCMCN /RJ/ ER,ERRL,ERRL,ETAC,ETAN,F
CCMCMCN /RJ/ FAR,G,GAMMA1,GAMMA2,GAMMA3,GAMMA4,GAMMA5,GAMMA6
CCMCMCN /RJ/ H,ISP,ISTAR,I1,I2,I3,I4,I5,I6,I7,I8,I9,I10
CCMCMCN /RJ/ CCUNT,ICCNV,ICCNVP,IFIRST
CCMCMCN /RJ/ M,M1,M2,M3,M4,M5,M6,MW,MW1,MW2,MW3,MW4,MW5,MW6
CCMCMCN /RJ/ MACR,MACRS,PM
CCMCMCN /RJ/ P,P1,P2,P3,P4,P5,P6,PT301,PRAR
CCMCMCN /RJ/ PT,PT1,PT2,PT3,PT4,PT5,PT6,PMOPI,PMOPI1,PMOPI2,PMOPI3
CCMCMCN /RJ/ PH1,PH2,PH3,PH4,PH5,PH6,PHI,PHI1,PHI2,PHI3,PHI4,PHI5,PHI6,PHI7,PHI8,PHI9,PHI10
CCMCMCN /RJ/ RADEG,R,SA,SART,I,SQRTOT
CCMCMCN /RJ/ T,T1,T2,T3,T4,T5,T6,TIME
CCMCMCN /RJ/ TT,TT1,TT2,TT3,TT4,TT5,TT6
CCMCMCN /RJ/ V,V1,V2,V3,V4,V5,V6
CCMCMCN /RJ/ WA,WB,X,Y,Z

```

```

PRESSURE RECCOVERY INSTALLED ALVJ -- BETA=0.0 (PS(I) I=15,81)

```

```

DATA FR/
A5.C,7.C,C,0,C,9,1,0,1,5,2,2,
A2.43,2,7,3,1,4,0,0,0,1,0,2,0,
A4.0,6.C,8,0,1,0,0,
A .5500,5475,9450,5400,9250,8775,8320,
A .9500,5475,9450,5400,9250,8775,8320,
A .5500,5475,9450,5400,9250,8775,8320,
A .5400,5375,9350,5300,9100,8600,8150,
A .7675,7660,7640,7600,7430,7000,6575,
A .6580,6860,6840,6800,6350,5750,5275,
A .5400,5375,5350,5300,4925,4300,3800,
A .3800,3760,3725,3650,3400,2850,2400,
A .2000,1960,1925,1850,1600,1050,0800/

```

```

AIR CAPTURE INSTALLED ALVRJ DEC 1975 AA(I) I=19,81

```

```

DATA AA/
A5.C,7.C,C,0,C,9,1,0,1,5,2,2,
A2.43,2,7,3,1,4,0,0,0,1,0,2,0,
A4.0,6.C,8,0,1,0,0,
A .5465,5457,9452,9440,9400,9350,9270,
A .5465,5457,9452,9440,9400,9350,9270,

```



```

C C C CCMFUTE CRITICAL PRESSURE RECOVERY AND MAXIMUM AIR CAPTURE RATIO
C
C     ALPHA=AE5(ALPHA)
C     CALL INTR20 (M,ALPHA,MACR,AA,1)
C     BLCT=CCL1(M,XM,BWR,12)
C     BLCTE=.015
C     BLEEC=ELCT5+BLDD8
C     MACRS=MACR*(1.0-BLEEC)
C     CALL INTR20 (M,ALPHA,PR1AD,PR,1)
C
C C C CRITICAL PRESSURE RECOVERY
C
C     PACR=PR1AD
C     IF (PRAR-PACR) 50,60,70
C
C C C PRESSURE RECOVERY .LT. CRITICAL
C
C     50 PT3OI=PRAR
C       ACR=1.
C       GC TC 80
C
C C C PRESSURE RECOVERY .EG. CRITICAL
C
C     60 PT3OI=PR1AD
C       ACR=1.
C       GC TC 8C
C
C C C PRESSURE RECOVERY .GT. CRITICAL --CHANGE AIR CAPTURE AND SET PRES
C     RECOVERY EQUAL TO CRITICAL VALUE
C
C     70 PT3OI=PR1AC
C       ACR=PR1AD/PRAR
C     80 POPT=FECPT(1.4,M)
C       PT=P/FCT
C       PT3=PT*PT3CI
C       PM=(PR1AC-PT3OI)/(PR1AC/100.)
C       AINF=ACR*AI*MACRS
C       SQTINF=SGRT(TT)
C       WA=(AINF*PT*FMOPT)/SQTINF
C       IF (ICGNVF.NE.0 .OR. ICCNV.EG.1) RETURN
C       CDSUB=C.0
C       IF (PRAR.LE.FACR) GC TC 50
C
C C C CALCULATE ADDITIVE DRAG DUE TO SUBSONIC SPILLAGE
C     ETASUB=M/((M*10.2095)-11.5652)
C     CDSUB=2.*((1.-ETASUB)*((1.-ACR)

```



```

C      CALCULATE ADDITIVE DRAG DUE TO SUPERSONIC SFILLAGE
C
90    CAED=CCL1(N,XM,CACD,12)
      IF (PRAR.LE.PACR) CD$UB=C.O
      CCASUB=CC$UP*AI/AR
      CCASUP=CAED
      CFB=C.C
C
      CFT IS RAT DRAG
      CFT=C.CC$2
      PESI=2.14678-1.11439*M+.22989*M*M
      FE=PESI*F
      AINFB=A1*BLCDB
      AE=9.33*.00064516
      TRN1=P*AE
      TRN2=P*AE
      WDCI=(FT*AINFB*PMOPT)/SQRTINF
      XMCE=WDCI*SCRT(TT-222.)/(PE*AE)
      XME=FRCM(1.4,XMCE)
      TRN3=PE*AE*(1.+1.4*XME*XME)
      CFC=-((TRN1+TRN2-TRN3)/(C*AR))
      RETURN
C
100   FORMAT (T6,'ALVRJ INLET DATA DEC 1975 ,TOL=',F7.6)
C
C      END
C
C      FUNCTION INTRF (AH,AL,BF,BL,FH,FM)
C
C      REAL INTRF
C2 = AL + (AH - AL)*FM
C1 = BL + (BF - BL)*FM
INTRF = C1 + (C2 - C1)*FT
RETURN
END
C
C      FUNCTION FATAS(GAM,XM)
C
CFATAS  FUNCTION SUBPRCGFAM TO CALCULATE A OVER ASTAR
        FRCM GAMMA AND MACH NUMBER
C
        Z=(GAM+1.C)/(2.0*(GAM-1.C))
        A=2.C/(GAM+1.0)
        B=(GAM-1.0)/2.0
        C=1.C/XM

```

```

FATA0030
FATA001C
FATA002C
FATA004C
FATA0050
FATA006C
FATA0070

```


FATA008C
FATA009C
FATA010C
FATA011C

R 2
R 4
R 6
R 8
R 10
R 12
R 14
R 16
R 18
R 20-

S 2
S 4
S 6
S 8
S 10
S 12
S 14
S 16
S 18
S 20-

FNAR0030
FNAR001C
FNAR002C
FNAR004C
FNAR005C
FNAR006C
FNAR007C
FNAR008C
FNAR009C
FNAR010C
FNAR011C

C=A*(1-C+(E**XN**XM))
FATAS=C*(C**Z)
RETURN
END

FUNCTION FIXAC (A)

10 TFIX=AES(A)-1.00000005
IF (TFIX.LT.0.0) GO TO 1C
IF (A.LT.0.0) GO TO 20
FIXAC=C
GO TO 3C
20 FIXAC=1.14159265
3C CONTINUE
RETURN
END

FUNCTION FIXAS (A)

10 TFIX=AES(A)-1.00000005
IF (TFIX.LT.0.0) GO TO 1C
IF (A.LT.0.0) GO TO 20
FIXAS=1.57079633
GO TO 3C
20 FIXAS=-1.57079633
3C CONTINUE
RETURN
END

FUNCTION FMARH (GAM,ATAS)

CFMARH FUNCTION SUBPROGRAM TO CALCULATE MACH NUMBER
FROM ATAS AND GAMMA

500 Z = (GAM + 1.0) / (2.0*(GAM - 1.0))
DXMG = 10.0
XMG = 10.0
XMG = EXMG / 2.0
XMG = AMAX1(1.0,AMIN1(100.0,XMG))
IF (DXMG - .000015)100,100,200
200 CATAS = 1.0/XMG*((2.0+(GAM - 1.0) *XMG **2)/(GAM + 1.0))**Z
IF (CATAS - ATAS)700,100,800


```

700 XMG = XMG + DXMG
    GO TO 500
800 XMG = XMG - DXMG
    GO TO 500
100 FMARL = XMG
    RETURN
    END

```

CC C C C C C

FUNCTION FMARL(GAM,AR)

FUNCTION ROUTINE TO FIND MACH FROM A/A* AND GAMMA
USING NEWTONS METHOD FOR FINDING ROOTS OF AN EQUATION

```

SUBSCNIC SCLUTION
IF(GAM.LT.1.) WRITE(6,20)
IF(GAM.LT.1.) WRITE(6,40)
IF(GAM.LT.1. .OR. AR.LT.1.) CALL EXIT
GMI=GAM-1.
GPI=GAM+1.
A=2./GFI
B=GMI/CF1
Z=GFI/(2.*GMI)
IF(AR.LT.1.3401) XMP=2.4706-(1.47059*AR)
IF(AR.GT.1.34 .AND. AR.LT.3.001) XMP=.77443- (.2048*AR)
IF(AF.GT.3.) XMP=(A*Z)/AR
TOL=.0001
10 CONTINUE
BRACK=A+(B*XMP*XMP)
POWER=BRACK**Z
TOP=(1./XMP)-(AR/POWER)
BOTTOM=(1./BRACK)-(1./XMP*XMP)
XM=XMP-(TOP/BOTTOM)
DELTA=ABS(XM-XMP)
XMP=XM
IF(DELTA.GT.TOL) GO TO 10
FMARL=XM
RETURN

```

```

30 FORMAT('GAMMA LT 1 IN FMARL')
40 FORMAT('A/A* LT 1 IN FMARL')
END

```

C C C C C C C

FMAR0120
FMAR0130
FMAR0140
FMAR0150
FMAR0160
FMAR0170
FMAR0180


```

C      FUNCTION FMPHIM (GAM , PHI )
C      CFMPHIM  FUNCTION SUBPRCGRAM TO CALC MACH NO  FROM PHIM
C
TOL = .CC001
XMSN = 1.0 / ( PHI**2 )
127 X = XMSN
XMSN = (1.0 + (GAM*(XMSN**2))) / (PHI * SQRT(2.0 * (GAM+1.0)*(1.0+
1+(GAM-1.0)*.5*(XMSN**2))))
XMSN = AMAX1(0., AMIN1(1.0,XMSN))
IF (ABS(X-XMSN)-TOL) 125, 125, 128
126 XMSN = (X + XMSN) / 2.0
GO TO 127
125 FMFHIM = XMSN
RETURN
END
C
C      FUNCTION FMPOPT (GAM,POPT)
C      CFMPOPT  FUNCTION SUBPRCGRAM TO CALC XMACH
C              FROM P/PT AND GAMMA
C
Z = (FCPT) ** ((1.0-GAM)/ GAM)
SMPPT = (Z - 1.0)/(GAM - 1.0) * 2.0
FMPCFT = SQRT(SMPPT)
RETURN
END
C
C      FUNCTION PHIM (GAM, XMCH)
C      PHIM  FUNCTION TO CALC PHI FROM GAMMA AND MACH
C
PHIM = (1.0+GAM *(XMCH **2)) / (XMCH * SQRT(2.0*(GAM + 1.0)*(1.0+
1.5 *(GAM -1.0) * (XMCH **2))))
RETURN
END
C
C      FUNCTION FPOPT (GAM,XMACH)
C      FPCPT  FUNCTION SUBPRCGRAM TO CALC TOTAL PRESSURE RATIO
C              FROM GAMMA AND MACH NO.

```



```

DATA A/, -4.013517C, -2.2874890, -2.1765372,
A-4.40382210, -1.3585554, .0355105, .0498002,
B-.0494300, .0061561, -.0030910, -.0121108,
C-.0055558, .0025277, -.0004221, .0044266,
D-.0026116, -.0039804, .0007523, -.0013126,
E-.0010261, .0033513, -.0005513, -.001147,
F-.0003149, -.0021286, .0002918, .0008523,
G-.0000031, .0010109, .99998983, 1.0000000/
H-.0001420, .0010109, .99998983, 1.0000000/

C ** SET CCNSTANTS
C
DATA PC,CC,RBAR,GC,Z1/
* 101325.0,1.225,287.07299,9.80665,30480.0/

C ** IF NECESSARY , CONVERT TO SI UNITS
C Z BECOMES METERS
C V BECOMES METERS/SEC

C
IF (K.EC.2) Z=Z*.3048
IF (K.EC.2.AND.KK.EC.2) V=V*.3048

C ** START HERE
C
IF (J) 10,20,20
1C WRITE (6,6C) Z1
WRITE (6,7C) Z1

C ** COMPUTE C(K) COEFFICIENTS
C
2C ETA=2.*(2.*Z/Z1-1.)
C(1)=ETA
C(2)=ETA**2-2.
DO 3C I=2,15
3C C(I)=ETA*C(I-1)-C(I-2)

C ** COMPLETE CHEBYSHEV TRUNCATED EXPANSION
C
ALNP=A(1,1)
ALND=A(2,1)
DO 4C I=2,15
4C ALNF=ALNP+A(1,I)*C(I-1)
ALND=ALND+A(2,I)*C(I-1)

C ** COMPUTE ATMOSPHERIC PRESSURE IN N/SQ METER
C
ALNP=.5*ALNP

```



```

      FOFZ=EXP(ALNP)
      F = FCF2*FC*A(1,16)
**      CCMPUTE GRAVITY AT ALTITUDE AND LATITUDE
      RC IS EQUATORIAL RADIUS (M) PER CRC HANDBOOK
      GC IN M/SEC SQ
      RC=6378377.45
      GO=GC*((RC/(RO+Z))**2)
**      CCMPUTE ATMOSPHERIC DENSITY
      D IN KG PER CUBIC METRE
      ALND=.5*ALNC
      CODZ=EXP(ALND)
      L = LCC2*CC*A(2,16)
**      CCMPUTE AIR TEMPERATURE
      T IN DEGREE KELVIN
      T=P/(REAR*D)
**      CCMPUTE SPEED OF SOUND IN M/SEC
      S=SQRT(401.90219*T)
**      CCMPUTE DYNAMIC PRESSURE IN PA
      IF (KK.EC.1) Q=0.7*F*V**2
      IF (KK.EC.2) Q=0.7*F*(V/5)**2
**      CONVERT TO ENGLISH UNITS IF NECESSARY
      IF (K.EC.1) GC TO 50
      Z=Z/.3048
      IF(KK.EC.2) V=V/.3048
      P=P/6894.757
      T=T*1.8
      S=S/.3048
      D=D/.1601846
      C=C/6894.757
      GC=GC/.3048
      RETURN
50  FORMAT (TS,'*** US 1962 STANDARD ATMOSPHERE')
60  FORMAT (TS,'*** ALTITUDE FROM 0 TO ',F8.0,' METERS')
7C  ENC
      CC

```



```

C
C
C
C
FUNCTION ANORM(A,B)
CALCULATES TOTAL PRESSURE RATIO ACROSS A NORMAL SHOCK
A=GAMMA,E=XNACH
C=(A+1.)/2.
C=(A-1.)/2.
E=(2.*A)/(A+1.)
F=(A-1.)/(A+1.)
G=A/(A-1.)
H=1./(A-1.)
BB=B*B
ANLN=((C*E)/(1.+C*E))*G)
DEM=((E*E-F)*H)
ANCRM = ANLN/DEM
RETURN
END

C
C
C
C
C
FUNCTION FMCPT( GAM , XM )
FUNCTION SUBPROGRAM TO CALCULATE P/PT * M CIRCLE
FROM GAMMA AND MACH NUMBER
DIMENSION S = SEC * SQRT(K) / METRE
FPMOPT = FMCIR ( GAM , XM ) * FPOPT ( GAM , XM )
RETURN
END

C
C
C
C
SUBROUTINE SERCH (X,A,J,I,M)
DIMENSION A(20)
DO 10 L=1,I
J=L
IF (X.LE.A(L)) GO TO 20
CONTINUE
20 IF (J.EG.1) GO TO 30
J=J-1
30 RETURN
END

C
C
C
C
FUNCTION SFCK(XM,D)

```

```

B 2
B 4
B 6
B 8
B 10
B 12
B 14
B 16
B 18
B 20-

```



```

SHCKC FUNCTION SUBPROGRAM TO CALCULATE THE OBLIQUE SHOCK ANGLE FROM
THE WEDGE ANGLE AND MACH NUMBER
XM=MACH NUMBER C=WEDGE ANGLE DELTA IN RADIANS
SHCKC=SHOCK ANGLE THETA IN RADIANS
P,Q,R,A,B=CONSTANTS F=PHI WHICH IS AN INTERMEDIATE ANGLE
RADEC=TRANSFORMS RADIANS TO DEGREES

```

```

RADEC = 57.2957795131
SINZCZ= SIN(C)
P= -((XM*XM+2.)/(XM*XM)) --(1.4*SINZCZ*SINZDZ)
CQ=1.44+(.4/(XM*XM))
C=((2.*XM*XM)+1.)/(XM*XM*XM*XM)+(CQ*SINZDZ*SINZDZ)
C=COS(C)
R=(-C*(C)/(XM*XM*XM*XM))
A=(1./3.)*(3.*Q-P*F)
B=(1./27.)*(2.*P*P*F-.5.*F*Q+.27.*R)
TEST=((B*B)/4.)+(A*A*A)/27.)
IF(TEST)1,1,2
1 CONTINUE
F=ARCCOS((-B/2.)/SQRT((-1./27.)*A*A*A))
FO=F*RADEC
FOX=(FC/3.)+240.
FOX=FCX/RADEC
X=2.*SQRT(-A/3.)*COS(FOX)
SHCKC= ARSIN(SQRT(X-P/3.))
RETURN
2 WRITE(6,3)
3 FORMAT(48F) FCW COME THE INPUTS TO SHOCK GIVE COMPLEX ROOTS)
RETURN
END

```

```

C 2
C 4
C 6
C 8
C 10
C 12
C 14
C 16
C 18
C 20
C 22
C 24
C 26
C 28

```

```

FUNCTION STOLI (TBX,TBY,TAB,X,Y,NX,NY)
FUNCTION DESIGNED TO ALLOW THE USER TO INPUT X
VARIABLE LENGTH TABLES. IT INTERPOLATES FIRST IN THE X
VARIABLE AND THEN IN THE Y VARIABLE. IT EXTRAPOLATES FROM
THE FIRST OR LAST INTERVAL IF THE X OR Y IS OUTSIDE OF THE
BOUNDS OF THE TABLE. IT REQUIRES THE SUBROUTINE SERCH FOR
ITS OPERATION.

```

```

PROGRAMMER -- F.SOBEL 30 OCTOBER 1967
DIMENSION TBX(NX), TBY(NY), TAB(11,11)
CALL SERCH (X,TBX,I,NX,1)
CALL SERCH (Y,TBY,J,NY,1)
XX=TBX(I)

```

```

C C C C C C C C C C

```



```

C      F1=-((.227276E-3+.100807E-6*T)*T+.018785)
C      F2=((C12C097-.1254E-5*T)*T+.0846043
C      F3=((C113312E-7-.24417E-11*T)*T-.141615E-4)*T+.00496295)*T+.0056
C      A0473
C
C      LINEARLY INTERPOLATE FOR FACTORS G1,G2,G3
C
C      DO 10 I=1,31
C      IF (ER.LE.E(I)) GO TO 20
C      CCNTINLE
C      J=I-1
C      R=(ER-E(J))/(E(I)-E(J))
C      G1=G(J,1)+(G(I,1)-G(J,1))*R
C      G2=G(J,2)+(G(I,2)-G(J,2))*R
C      G3=G(J,3)+(G(I,3)-G(J,3))*R
C
C      LINEARLY INTERPOLATE FOR FACTORS H1,H2
C
C      DO 30 I=1,13
C      IF (ATM.LE.P(I)) GO TO 40
C      CCNTINLE
C      J=I-1
C      R=(ATM-P(J))/(P(I)-P(J))
C      H1=H(J,1)+(H(I,1)-H(J,1))*R
C      H2=H(J,2)+(H(I,2)-H(J,2))*R
C
C      COMPLETE TEMPERATURE CORRECTION (TC)
C
C      TC=C(1)*F1*G1*H1+C(2)*F2*G2*H1+C(3)*F1*G2*H2+C(4)*F2*G1*H2
C      D      + C(5)*F3*G3*F1
C      E      + C(6)*F2*G3*H2
C      F      + C(7)*F3*G2*H2
C      G      + C(8)*F1*G3*H2
C
C      RETURN (6,110)
C      GO TO 100
C      50 WRITE (6,120)
C      GO TO 100
C      60 WRITE (6,130)
C      GO TO 100
C      70 WRITE (6,140)
C      GO TO 100
C      80 WRITE (6,150)
C      GO TO 100
C      90 WRITE (6,160)
C      CALL EXIT
C      100 RETURN
C

```



```

CC 1C 11=4,IX1
I=11
IF(X.LT.C(I)) GO TO 11
CONTINUE
10 IO=I-2
11 IX2=IX1+4
IX3=IX1+N2
DO 2C JJ=IX2,IX3
J=JJ
IF(Y.LT.C(J)) GO TO 21
CONTINUE
20 JO=J-1X1
21 NA=IX3+N2*((L-1)*N1+IC-2)+JO-1
F1=(X-C(I-1))/(O(I)-C(I-1))
F2=(Y-C(J-1))/(O(J)-C(J-1))
F3=F1*F2
C=C(NA)*(1-F1-F2+F3)
1 + C(NA+N2)*(F1-F3)
2 + C(NA+1)*(F2-F3)
3 + C(NA+N2+1)*F3
RETURN
END

```

CC C C C C C C

FUNCTION THREDL (X,Y,Z,AX,AY,AZ,XYZ,NX,NY,NZ)

THIS FUNCTION INTERPOLATES LINEARLY IN THREE DIMENSIONS. IT
 SELECTS THE TWO PLANES OF CONSTANT Z AND USES STDLIA FOR TWO DIMEN-
 SIONAL INTERPOLATION. A LINEAR INTERPOLATION IS USED IN Z.
 SIGNAL INTERPOLATION. H. SOBDEL
 PROGRAMMER
 DIMENSION AX(NX), AY(NY), AZ(NZ), XYZ(NX,NY,NZ)
 DATA M/1/
 CALL SEFCF (Z,AZ,K,NZ,M)
 AL=STDLIA(AX,AY,XYZ(1,1,K),X,Y,NX,NY)
 AU=STDLIA(AX,AY,XYZ(1,1,K+1),X,Y,NX,NY)
 THREDL=AL+(AU-AL)*(Z-AZ(K))/(AZ(K+1)-AZ(K))
 RETURN
 END

F	2	
F	4	
F	6	
F	8	
F	10	
F	12	
F	14	
F	16	
F	18	
F	20	
F	22	
F	24	
F	26	-

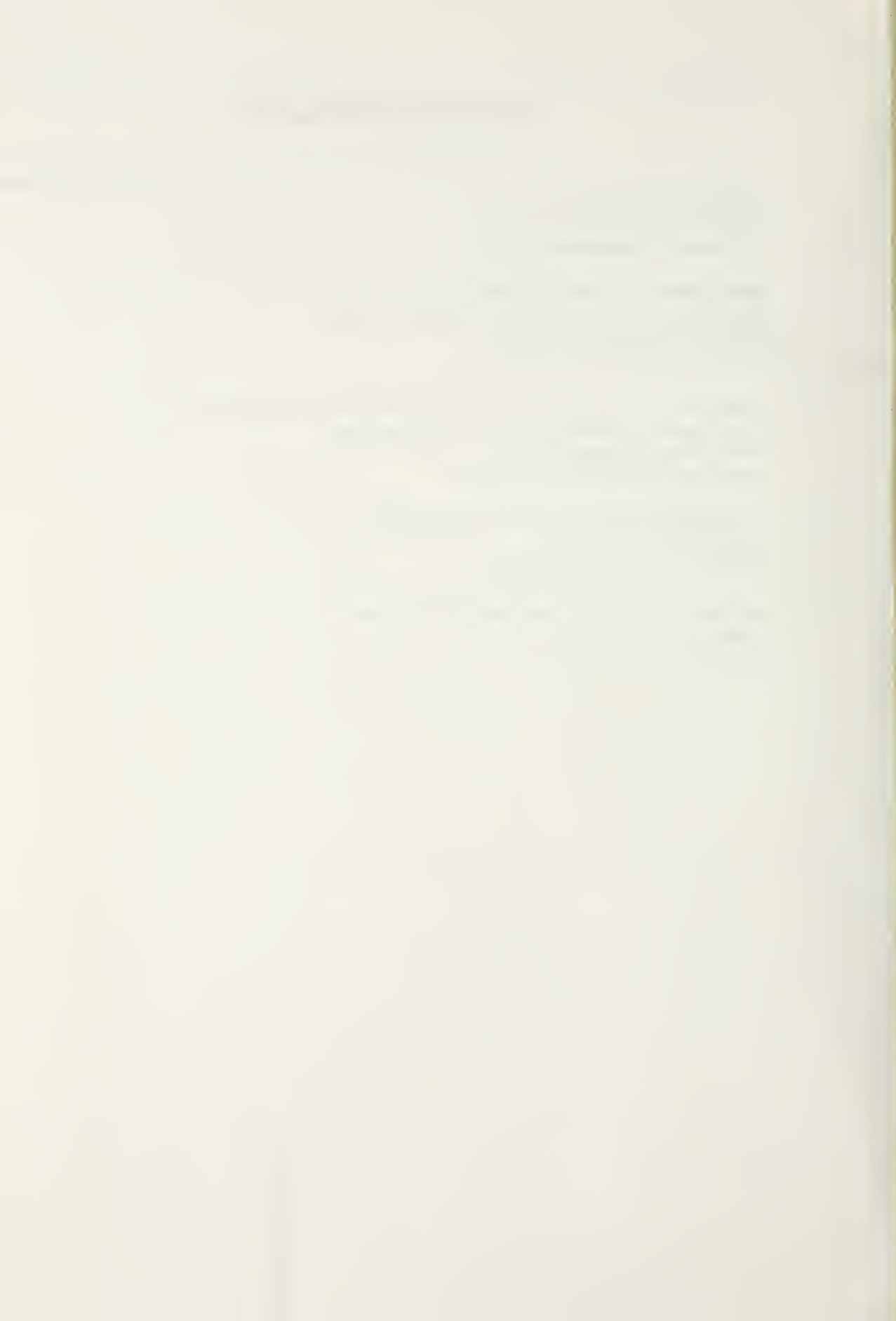
LIST OF REFERENCES

1. Honeywell Defense System Division, System Design Tradeoff Study For Supersonic Tactical Missile (STM), Volume I, by Robert H. Josselson, p. 9-17, 24 March 1978.
2. Honeywell Defense System Division, System Design Tradeoff Study For Supersonic Tactical Missile (STM), Volume I, by Robert H. Josselson, p. 11, 24 March 1978.
3. Honeywell Defense System Division, System Design Tradeoff Study For Supersonic Tactical Missile (STM), Volume I, by Robert H. Josselson, p. 21, 24 March 1978.
4. Litton Systems, Inc., Guidance and Control Systems Division, Modularized Six-Degree-Of-Freedom (MOD6DF) Computer Program, Volume I, by John W. Diesel and others, p. 1-1.
5. Litton Systems, Inc., Guidance and Control Systems Division, Modularized Six-Degree-Of-Freedom (MOD6DF) Computer Program, Volume II, by John W. Diesel and others, p. C-1-C-37.
6. Litton Systems, Inc., Guidance and Control Systems Division, Modularized Six-Degree-Of-Freedom (MOD6DF) Computer Program, Volume II, by John W. Diesel and others, p. C-38-C-51.
7. Litton Systems, Inc., Guidance and Control Systems Division, Modularized Six-Degree-Of-Freedom (MOD6DF) Computer Program, Volume I, by John W. Diesel and others, p. 3-18.
8. Litton Systems, Inc., Guidance and Control Systems Division, Modularized Six-Degree-Of-Freedom (MOD6DF) Computer Program, Volume I, by John W. Diesel and others, p. 3-16.
9. Litton Systems, Inc., Guidance and Control Systems Division, Modularized Six-Degree-Of-Freedom (MOD6DF) Computer Program, Volume I, by John W. Diesel and others, p. 3-37.
10. Litton Systems, Inc., Guidance and Control Systems Division, Modularized Six-Degree-Of-Freedom (MOD6DF) Computer Program, Volume I, by John W. Diesel and others, p. 3-10.
11. Naval Postgraduate School Technical Note 0141-25, User's Guide to Programming Aids Library, by Lloyd G. Nolan, p. 27-29, October 1978.
12. Honeywell Defense Systems Division, System Design Tradeoff Study For Supersonic Tactical Missile (STM), Volume I, by Robert H. Josselson, 24 March 1978.

13. Honeywell Defense Systems Division, Systems Design Tradeoff Study For Supersonic Tactical Missile (STM), Volume I, by Robert H. Josselson, p. 77-82, 24 March 1978.
14. Litton Systems, Inc., Guidance and Control Systems Division, Modularized Six-Degree-Of-Freedom (MOD6DF) Computer Program, Volume I, by John W. Diesel and others, p. 4-41-4-83.
15. Litton Systems, Inc., Guidance and Control Systems Division, Modularized Six-Degree-Of-Freedom (MOD6DF) Computer Program, Volume II, by John W. Diesel and Others, p. B-1 - B-49.

INITIAL DISTRIBUTION LIST

	No. Copies
1. Library, Code 0142 Naval Postgraduate School Monterey, California 93940	2
2. Department Chairman, Code 67 Department of Aeronautical Engineering Naval Postgraduate School Monterey, California 93940	1
3. Professor D. J. Collins, Code 67C0 (thesis advisor) Department of Aeronautical Engineering Naval Postgraduate School Monterey, California 93940	2
4. LT Thomas Alan Grote, USN (student) 1016 Orchid Avenue Modesto, California 95355	1
5. Defense Technical Information Center Cameron Station Alexandria, Virginia 22314	2



Thesis
G86075
c.1

Grote

Guidance and control
of tactical missiles.

188001

20 APR 83
20 APR 83
OCT 3 85

29416
33039

8801

control
files.

39

Thesis
G86075
c.1

Grote

Guidance and control
of tactical missiles.

188001

thesG86075

Guidance and control of tactical missile



3 2768 002 13568 3

DUDLEY KNOX LIBRARY